

Biometric-Based Efficient & Secure Access Mechanism for Cloud Services

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology degree in Information Technology

By

VEMANA AKHIL NEERAJ (37120083)

SAI ASHIK PERNI (37120053)



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

MARCH - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Vemana Akhil Neeraj (37120083)**, **Sai Ashik Perni (37120053)** who carried out the project entitled **"Biometric Based Efficient and Secure Access Mechanism For Cloud Services"** under my supervision from December 2020 to March 2021.

Internal Guide
Mr. P. Saravanan M.E.,

Head of the Department
Dr. R SUBHASHINI M.E, Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I , **Vemana Akhil Neeraj (37120083)**, **Sai Ashik Perni (37120053)** here by declare that the Project Report entitled “**Biometric Based Efficient and Secure Access Mechanism For Cloud Services**” done by me under the guidance of **Mr.P.Saravanan M.E.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering / Technology degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THECANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing , **Dr.R.Subhashini M.E., Ph.D. , Head** of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mr.P.Saravanan M.E.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

The demand for remote data storage and computation services is increasing exponentially in our data-driven society; thus, the need for secure access to such data and services. In this paper, we design a new biometric-based authentication protocol to provide secure access to a remote (cloud) server. In the proposed approach, we consider biometric data of a user as a secret credential. We then derive a unique identity from the user's biometric data, which is further used to generate the user's private key. In addition, we propose an efficient approach to generate a session key between two communicating parties using two biometric templates for a secure message transmission. In other words, there is no need to store the user's private key anywhere and the session key is generated without sharing any prior information. A detailed Real-Or- Random (ROR) model based formal security analysis, informal (non-mathematical) security analysis and also formal security verification using the broadly-accepted Automated Validation of Internet Security Protocols and Applications (AVISPA) tool reveal that the proposed approach can resist several known attacks against (passive/active) adversary. Finally, extensive experiments and a comparative study demonstrate the efficiency and utility of the proposed approach.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE.NO
	LIST OF FIGURES	VII
	LIST OF ABBREVIATIONS	IX
1	INTRODUCTION	
	1.1 About the Project	8
2	SYSTEM ANALYSIS	
	2.1 Existing system	9
	2.2 Proposed system	9
3	REQUIREMENTS SPECIFICATION	
	3.1 Introduction	11
	3.2 Hardware and Software specification	12
	3.3 Technologies Used	12
	3.4 Java	13
	3.5 Introduction to Java	13
	3.6 [Working of Java	14
4	Project Purpose and Scope	
	4.1 Purpose	27
	4.2 Project Scope	27
	4.3 Product Perspective	27

	4.4 System Features	28
	4.5 Design and Implementation Constraints	28
	4.6 Other Nonfunctional Requirements	29
5	SYSTEM DESIGN	
	5.1 Architecture Diagram	31
	5.2 Sequence Diagram	32
	5.3 Use Case Diagram	32
	5.4 Activity Diagram	34
	5.5 Collaboration Diagram	35
	5.6 DATA FLOW DIAGRAM	35
	5.7 Class Diagram	37
6	SYSTEM DESIGN – DETAILED	
	6.1 Modules	38
	6.2 Module explanation	38
7	CODING AND TESTING	
	7.1 Coding	40

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO
1	JVM	8
2	Java interpreter	9
3	Java platform	10
4	cloud	13
5	Virtualized cloud model	16
6	Service-oriented Architecture(SOA)	17
7	Grid Computing	18
8	System Design	24
9	Sequence Diagram	25
10	Use Case Diagram	26
11	Activity Diagram	27
12	Collaboration Diagram:	28
13	Data Flow Diagram	29
14	Class diagram	30

LIST OF ABBREVIATION

ABBREVIATION	EXPANSION
JDK	Java Development Toolkit
DEX	Dalvik Executables
TCP	Transmission Control Protocol
IP	Internet Protocol
HTTP	Hyper Text Transfer Protocol
ADT	Android Development Tool

CHAPTER 1

INTRODUCTION

Aim:

The main aim of the project is used to derive a unique identity from the user's biometric data, which is further used to generate the user's private key.

Synopsis:

The demand for remote data storage and computation services is increasing exponentially in our data-driven society; thus, the need for secure access to such data and services. In this paper, we design a new biometric-based authentication protocol to provide secure access to a remote (cloud) server. In the proposed approach, we consider biometric data of a user as a secret credential. We then derive a unique identity from the user's biometric data, which is further used to generate the user's private key. In addition, we propose an efficient approach to generate a session key between two communicating parties using two biometric templates for a secure message transmission. In other words, there is no need to store the user's private key anywhere and the session key is generated without sharing any prior information. A detailed Real-OrRandom (ROR) model based formal security analysis, informal (non-mathematical) security analysis and also formal security verification using the broadly-accepted Automated Validation of (AVISPA) tool reveal that the proposed approach can resist several known attacks against (passive/active) adversary. Finally, extensive experiments and a comparative study demonstrate the efficiency and utility of the proposed approach.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

In existing system uploading and sharing file among multiple client user in cloud environment is very hard to perform and there is no proper authentication among the cloud user and to the cloud server. Hence the file that uploaded on the cloud would not be secure as there is a lot of security problem that related to the cloud storage. In the authentication phase, we capture a new biometric fingerprint image of the user, and subsequently generate the private key and encrypt the biometric data as a query. This queried biometric data is then transmitted to the authentication server for matching with the stored data

Problem Definition

File sharing among multiple Clients is not secure. To generate a revocable private key directly from an irrevocable fingerprint image. There is no need to store the private key or a direct form of the user's biometric data anywhere.

2.2 PROPOSED SYSTEM

In Proposed system, we introduced secure and efficient file storage and sharing cloud environment using Remote cloud. Mutual authentication is very important between two target device as the user and to the cloud server as mutual authentication makes the server to trust the cloud storage where cloud owner can store the data and from server end, server will verify the user credential to provide any service to that particular user. Cloud contains the encrypted information along with the parameter related to that file. If any user request to the cloud server with some attribute. Based on the attribute cloud will redirect all the related file to the user in order to decrypt the file user need the key for that file for that user request to RSA and DSE(Key Generation Centre) with the file attribute

Advantage

Biometric has its unique advantages over conventional password and token-based security system, as evidenced by its increased adoption

CHAPTER 3

REQUIREMENT SPECIFICATIONS

3.1 INTRODUCTION

Cloud services are a norm in our society. However, providing secure access to cloud services is not a trivial task, and designing robust authentication, authorization and accounting for access is an ongoing challenge, both operationally and research-wise. A number of authentication mechanisms have been proposed in the literature, such as those based on Kerberos [1], OAuth [2] and OpenID [3] (see [1], [4]– [12]). Generally, these protocols seek to establish a secure delegated access mechanism among two communicating entities connected in a distributed system. These protocols are based on the underlying assumption that the remote server responsible for authentication is a trusted entity in the network. Specifically, a user first registers with a remote server. This is needed to ensure the authorization of the owner. When a user wishes to access a server, the remote server authenticates the user and the user also authenticates the server. Once both verifications are successfully carried out, the user obtains access to the services from some remote server. One key limitation in existing authentication mechanisms is that the user's credentials are stored in the authentication server, which can be stolen and (mis)used to gain unauthorized access to various services. Also, to ensure secure and fast communication, existing mechanisms generally use symmetric key cryptography, which requires a number of cryptographic keys to be shared during the authentication process. This strategy results in an overhead to the authentication protocols. Designing secure and efficient authentication protocols is challenging, as evidenced by the weaknesses revealed in the published protocols of Jiang et al. [13], Althobaiti et al. [14], Xue et al. [15], Turkanovic et al. [16], Park et al. [17], Dhillon and Kalra [18], Kaul and Awasthi [19] and Kang et al. [20] – see also Section II. Therefore, in this paper we seek to design a secure and efficient authentication protocol. Specifically, we will first provide an alternative to conventional password-based authentication mechanism. Then, we demonstrate how one can build a secure communication between communicating parties involved in the authentication protocol, without having any secret pre-loaded (i.e., shared) information. In the proposed approach, we consider a fingerprint image of a user as a secret credential. From the fingerprint image, generate a private key that is used to enroll the user's

credential secretly in the database of an authentication server. In the authentication phase, we capture a new biometric fingerprint image of the user, and subsequently generate the private key and encrypt the biometric data as a query. This biometric data is then transmitted to the authentication server for matching with the stored data. Once the user is authenticated successfully, he/she is ready to access his/her service from the desired server. To obtain secure access to the service server, mutual authentication between the user and authentication server, and also between the user and service server have been proposed using a short-term session key. Using two fingerprint data, we present a fast and robust approach to generate the session key. In addition, a biometric-based message authenticator is also generated for message authenticity purpose.

3.2 HARDWARE AND SOFTWARE SPECIFICATION

3.2.1 HARDWARE REQUIREMENTS

- Hard Disk : 250GB and Above
- RAM : 4GB and Above
- Processor : i3 and Above

3.2.2 SOFTWARE REQUIREMENTS

- Hard Disk : 250GB and Above
- RAM : 4GB and Above
- Processor : i3 and Above

3.3 TECHNOLOGIES USED

- J2EE(JAVA,JSP,JSTL,JSON)
- JavaScript , HTML , CSS
- Spring MVC

3.3.1 JAVA

Java is an object-oriented programming language developed initially by James Gosling and colleagues at Sun Microsystems. The language, initially called Oak (named after the oak trees outside Gosling's office), was intended to replace C++, although the feature set better resembles that of Objective C.

3.3.1.1 INTRODUCTION TO JAVA

Java has been around since 1991, developed by a small team of Sun Microsystems developers in a project originally called the Green project. The intent of the project was to develop a platform-independent software technology that would be used in the consumer electronics industry. The language that the team created was originally called Oak.

The first implementation of Oak was in a PDA-type device called Star Seven (*7) that consisted of the Oak language, an operating system called GreenOS, a user interface, and hardware. The name *7 was derived from the telephone sequence that was used in the team's office and that was dialed in order to answer any ringing telephone from any other phone in the office.

Around the time the First Person project was floundering in consumer electronics, a new craze was gaining momentum in America; the craze was called "Web surfing." The World Wide Web, a name applied to the Internet's millions of linked HTML documents was suddenly becoming popular for use by the masses. The reason for this was the introduction of a graphical Web browser called Mosaic, developed by ncSA. The browser simplified Web browsing by combining text and graphics into a single interface to eliminate the need for users to learn many confusing UNIX and DOS commands. Navigating around the Web was much easier using Mosaic.

It has only been since 1994 that Oak technology has been applied to the Web. In 1994, two Sun developers created the first version of Hot Java, and then called Web Runner, which is a graphical browser for the Web that exists today. The browser was coded entirely in the Oak language, by this time called Java. Soon after, the Java compiler was rewritten in the Java

language from its original C code, thus proving that Java could be used effectively as an application language. Sun introduced Java in May 1995 at the Sun World 95 convention.

Web surfing has become an enormously popular practice among millions of computer users. Until Java, however, the content of information on the Internet has been a bland series of HTML documents. Web users are hungry for applications that are interactive, that users can execute no matter what hardware or software platform they are using, and that travel across heterogeneous networks and do not spread viruses to their computers. Java can create such applications.

3.3.1.2 WORKING OF JAVA

For those who are new to object-oriented programming, the concept of a class will be new to you. Simplistically, a class is the definition for a segment of code that can contain both data (called attributes) and functions (called methods).

When the interpreter executes a class, it looks for a particular method by the name of **main**, which will sound familiar to C programmers. The main method is passed as a parameter an array of strings (similar to the argv [] of C), and is declared as a static method.

To output text from the program, we execute the **println** method of **System.out**, which is java's output stream. UNIX users will appreciate the theory behind such a stream, as it is actually standard output. For those who are instead used to the Wintel platform, it will write the string passed to it to the user's program.

Java consists of two things :

- Programming language
- Platform

3.3.1.3 THE JAVA PROGRAMMING LANGUAGE

Java is a high-level programming language that is all of the following:

- Simple

- Object-oriented
- Distributed
- Interpreted
- Robust
- Secure
- Architecture-neutral
- Portable
- High-performance
- Multithreaded
- Dynamic

The code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

Java is unusual in that each Java program is both compiled and interpreted. With a compiler, you translate a Java program into an intermediate language called **Java byte codes** – the platform independent codes interpreted by the Java interpreter. With an interpreter, each Java byte code instruction is parsed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. This figure illustrates how it works:

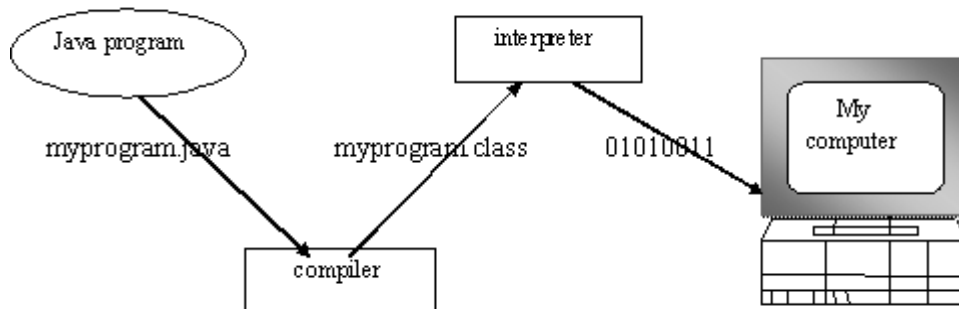


Fig.1

You can think of Java byte codes as the machine code instructions for the **Java Virtual Machine (JVM)**. Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of JVM. That JVM can also be implemented in hardware. Java byte codes help make “write once, run anywhere” possible.

You can compile your Java program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the JVM. For example, that same Java program can e run on Windows NT, Solaris and Macintos

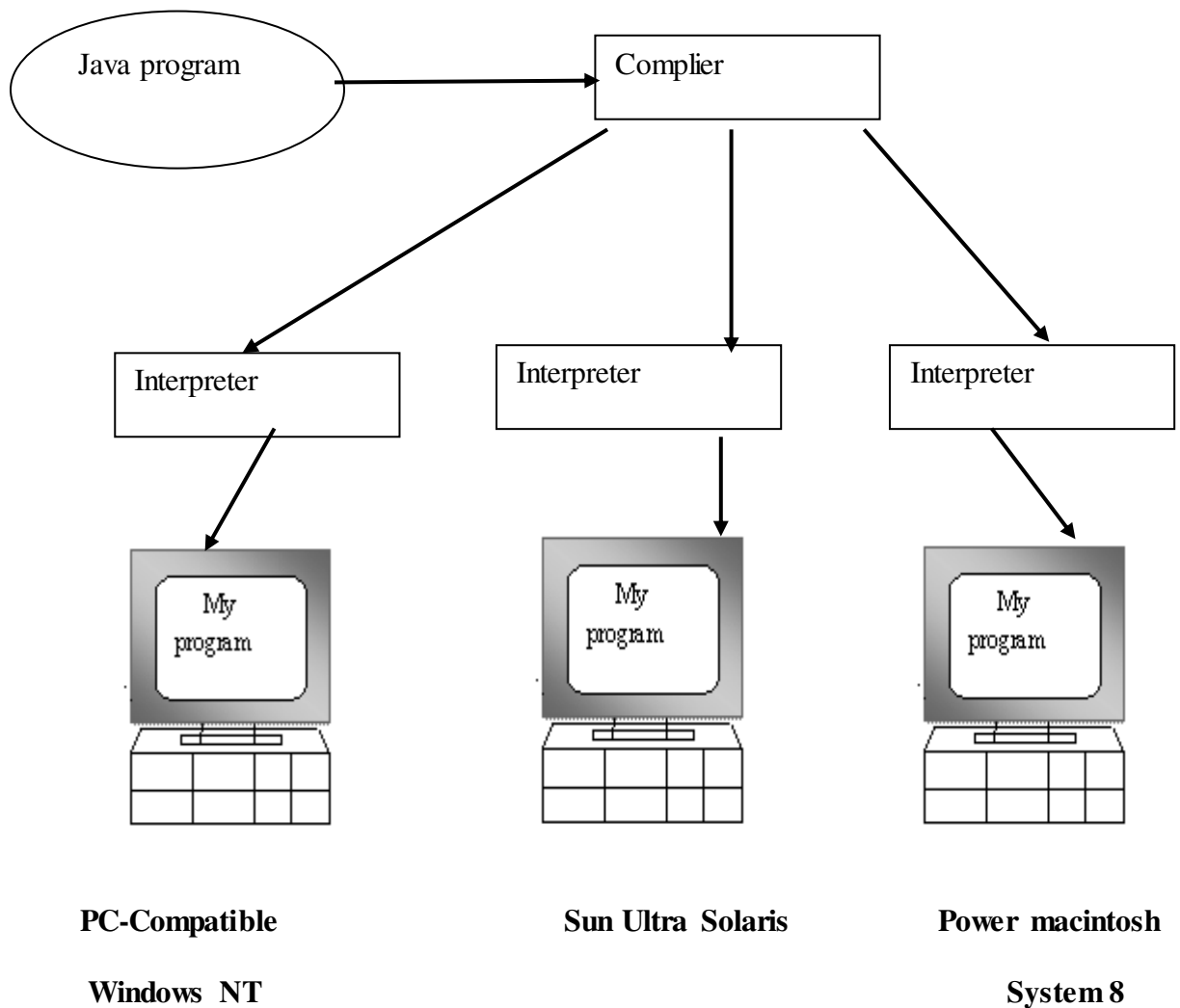


Fig.2

3.3.1.4 THE JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components :

- The Java Virtual Machine (JVM)
- The Java Application Programming Interface (Java API)

You've already been introduced to the JVM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries (**packages**) of related components. The following figure depicts a Java program, such as an application or applet, that's running on the Java platform. As the figure shows, the Java API and Virtual Machine insulates the Java program from hardware dependencies.

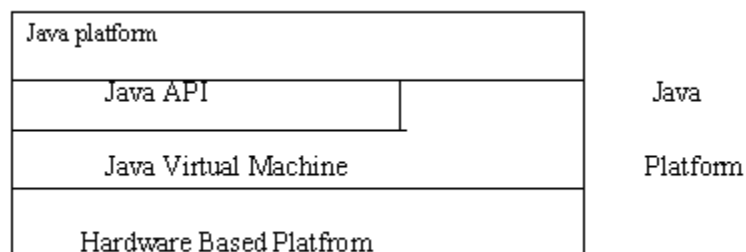


Fig.3

As a platform-independent environment, Java can be a bit slower than native code. However, smart compilers, weel-tuned interpreters, and just-in-time byte compilers can bring Java's performance close to that of native code without threatening portability.

3.3.2 APACHE TOMCAT SERVER

Apache Tomcat (formerly under the Apache Jakarta Project; Tomcat is now a top level project) is a web container developed at the Apache Software Foundation. Tomcat implements the servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server. It adds tools for configuration and management but can also be configured by editing configuration files that are normally XML-formatted. Because Tomcat includes its own HTTP server internally, it is also considered a standalone web server.

Environment

Tomcat is a web server that supports servlets and JSPs. Tomcat comes with the Jasper compiler that compiles JSPs into servlets.

The Tomcat servlet engine is often used in combination with an Apache web server or other web servers. Tomcat can also function as an independent web server. Earlier in its development, the perception existed that standalone Tomcat was only suitable for development environments and other environments with minimal requirements for speed and transaction handling. However, that perception no longer exists; Tomcat is increasingly used as a standalone web server in high-traffic, high-availability environments.

Since its developers wrote Tomcat in Java, it runs on any operating system that has a JVM.

Product features

Tomcat 3.x (initial release)

- implements the Servlet 2.2 and JSP 1.1 specifications
- servlet reloading

- basic HTTP functionality Tomcat 4.x
- implements the Servlet 2.3 and JSP 1.2 specifications
- servlet container redesigned as Catalina
- JSP engine redesigned as Jasper
- Coyote connector
- Java Management Extensions (JMX), JSP and Struts-based administration
- Tomcat 5.x
- implements the Servlet 2.4 and JSP 2.0 specifications
- reduced garbage collection, improved performance and scalability
- native Windows and Unix wrappers for platform integration
- faster JSP parsing

History

Tomcat started off as a servlet specification implementation by James Duncan Davidson, a software architect at Sun. He later helped make the project open source and played a key role in its donation by Sun to the Apache Software Foundation.

Davidson had initially hoped that the project would become open-sourced and, since most open-source projects had O'Reilly books associated with them featuring an animal on the cover, he wanted to name the project after an animal. He came up with Tomcat since he reasoned the animal represented something that could take care of and fend for itself. His wish to see an animal cover eventually came true when O'Reilly published their Tomcat book with a tomcat on the cover.

3.3.3 Introduction to Cloud Computing

Cloud Computing provides us means of accessing the applications as utilities over the Internet. It allows us to create, configure, and customize the applications online.

The term Cloud refers to a Network or Internet. In other words, we can say that Cloud is something, which is present at remote location. Cloud can provide services over public and private networks, i.e., WAN, LAN or VPN.

Applications such as e-mail, web conferencing, customer relationship management (CRM) execute on cloud. Cloud Computing refers to manipulating,

configuring, and accessing the hardware and software resources remotely. It offers online data storage, infrastructure, and application.

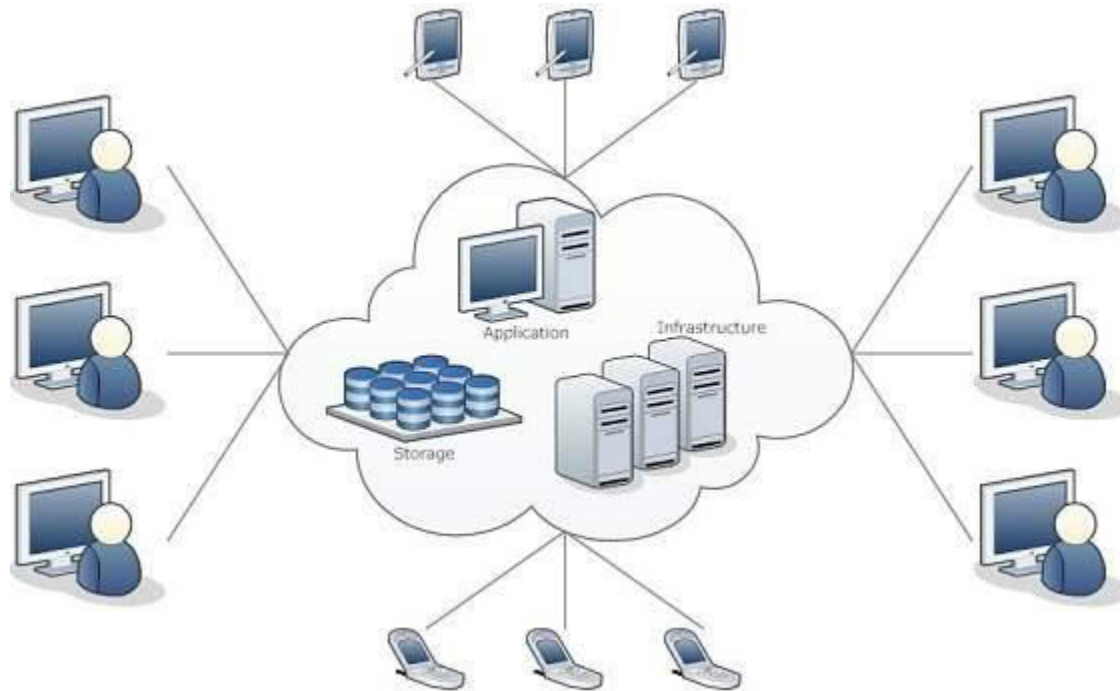


Fig.4

Cloud computing offers platform independency, as the software is not required to be installed locally on the PC. Hence, the Cloud Computing is making our business applications mobile and collaborative.

There are certain services and models working behind the scene making the cloud computing feasible and accessible to end users. Following are the working models for cloud computing:

- Deployment Models
- Service Models

Cloud Service Models

Infrastructure-as-a-Service provides access to fundamental resources such as physical machines, virtual machines, virtual storage, etc. Apart from these resources, the IaaS also offers:

- Virtual machine disk storage
- Virtual local area network (VLANs)

- Load balancers
- IP addresses
- Software bundles

All of the above resources are made available to end user via server virtualization. Moreover, these resources are accessed by the customers as if they own them.

Platform-as-a-Service offers the runtime environment for applications. It also offers development and deployment tools required to develop applications. PaaS has a feature of point-and-click tools that enables non-developers to create web applications.

App Engine of Google and Force.com are examples of PaaS offering vendors. Developer may log on to these websites and use the built-in API to create web-based applications.

But the disadvantage of using PaaS is that, the developer locks-in with a particular vendor. For example, an application written in Python against API of Google, and using App Engine of Google is likely to work only in that environment.

The following diagram shows how PaaS offers an API and development tools to the developers and how it helps the end user to access business applications.

Software-as-a-Service (SaaS) model allows providing software application as a service to the end users. It refers to software that is deployed on a host service and is accessible via Internet.

There are several SaaS applications listed below:

- Billing and invoicing system
- Customer Relationship Management (CRM) applications
- Help desk applications
- Human Resource (HR) solutions

Some of the SaaS applications are not customizable such as Microsoft Office Suite. But SaaS provides us Application Programming Interface (API), which allows the developer to develop a customized application.

Identity-as-a-Service (IDaaS) Employees in a company require to login to system to perform various tasks. These systems may be based on local server or cloud based. Following are the problems that an employee might face:

- Remembering different username and password combinations for accessing multiple servers.
- If an employee leaves the company, it is required to ensure that each account of that user is disabled. This increases workload on IT staff.

To solve above problems, a new technique emerged which is known as **Identity-as-a-Service (IDaaS)**.

IDaaS offers management of identity information as a digital entity. This identity can be used during electronic transactions.

Network-as-a-Service allows us to access to network infrastructure directly and securely. NaaS makes it possible to deploy custom routing protocols.

NaaS uses virtualized network infrastructure to provide network services to the customer. It is the responsibility of NaaS provider to maintain and manage the network resources. Having a provider working for a customer decreases the workload of the customer. Moreover, NaaS offers network as a utility. NaaS is also based on pay-per-use model.

How NaaS is delivered?

To use NaaS model, the customer is required to logon to the web portal, where he can get online API. Here, the customer can customize the route.

In turn, customer has to pay for the capacity used. It is also possible to turn off the capacity at any time.

Mobile NaaS

Mobile NaaS offers more efficient and flexible control over mobile devices. It uses virtualization to simplify the architecture thereby creating more efficient processes.

Cloud Computing Technologies

There are certain technologies working behind the cloud computing platforms making cloud computing flexible, reliable, and usable. These technologies are listed below:

- Virtualization
- Service-Oriented Architecture (SOA)
- Grid Computing
- Utility Computing

Virtualization

Virtualization is a technique, which allows sharing single physical instance of an application or resource among multiple organizations or tenants (customers). It does this by assigning a logical name to a physical resource and providing a pointer to that physical resource when demanded.

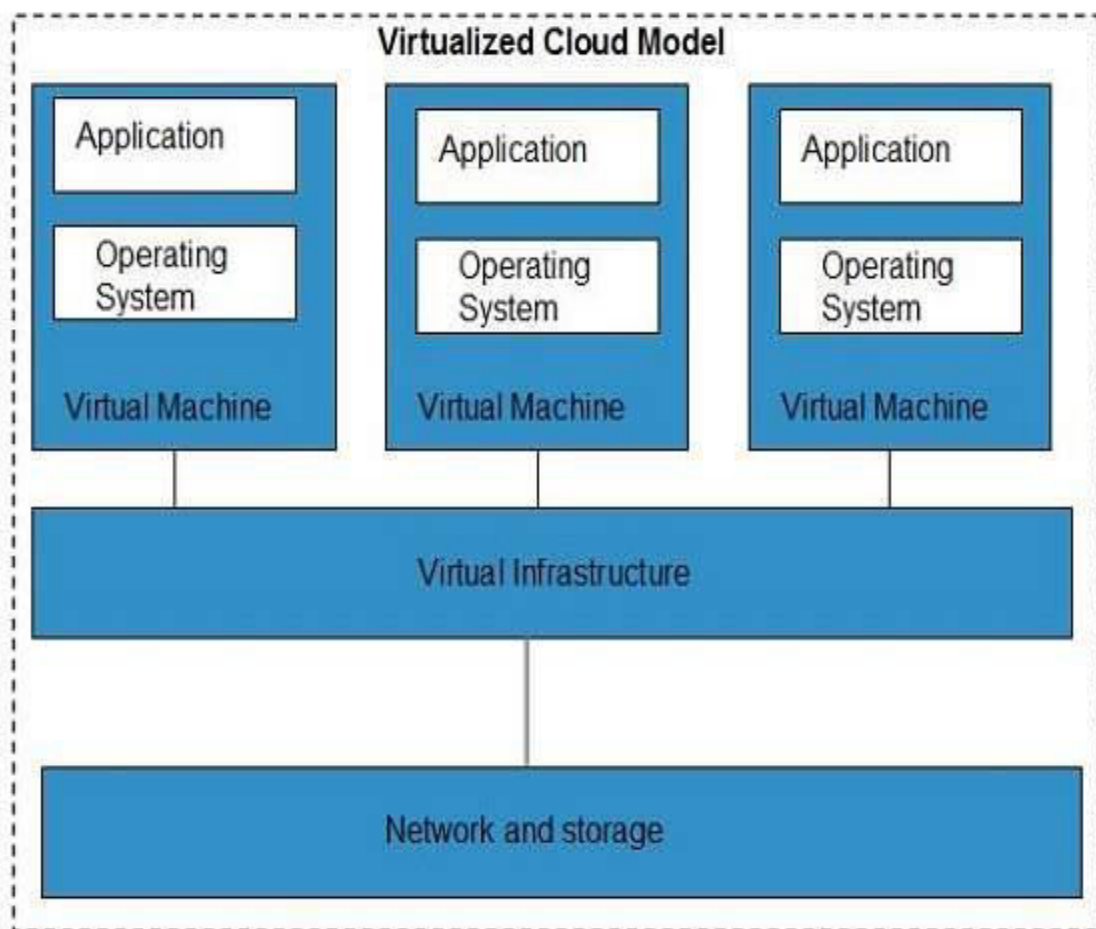


Fig.5

The **Multitenant** architecture offers **virtual isolation** among the multiple tenants. Hence, the organizations can use and customize their application as though they each have their instances running.

Service-Oriented Architecture (SOA)

Service-Oriented Architecture helps to use applications as a service for other applications regardless the type of vendor, product or technology. Therefore, it is possible to exchange the data between applications of different vendors without additional programming or making changes to services.

The cloud computing service oriented architecture is shown in the diagram below.

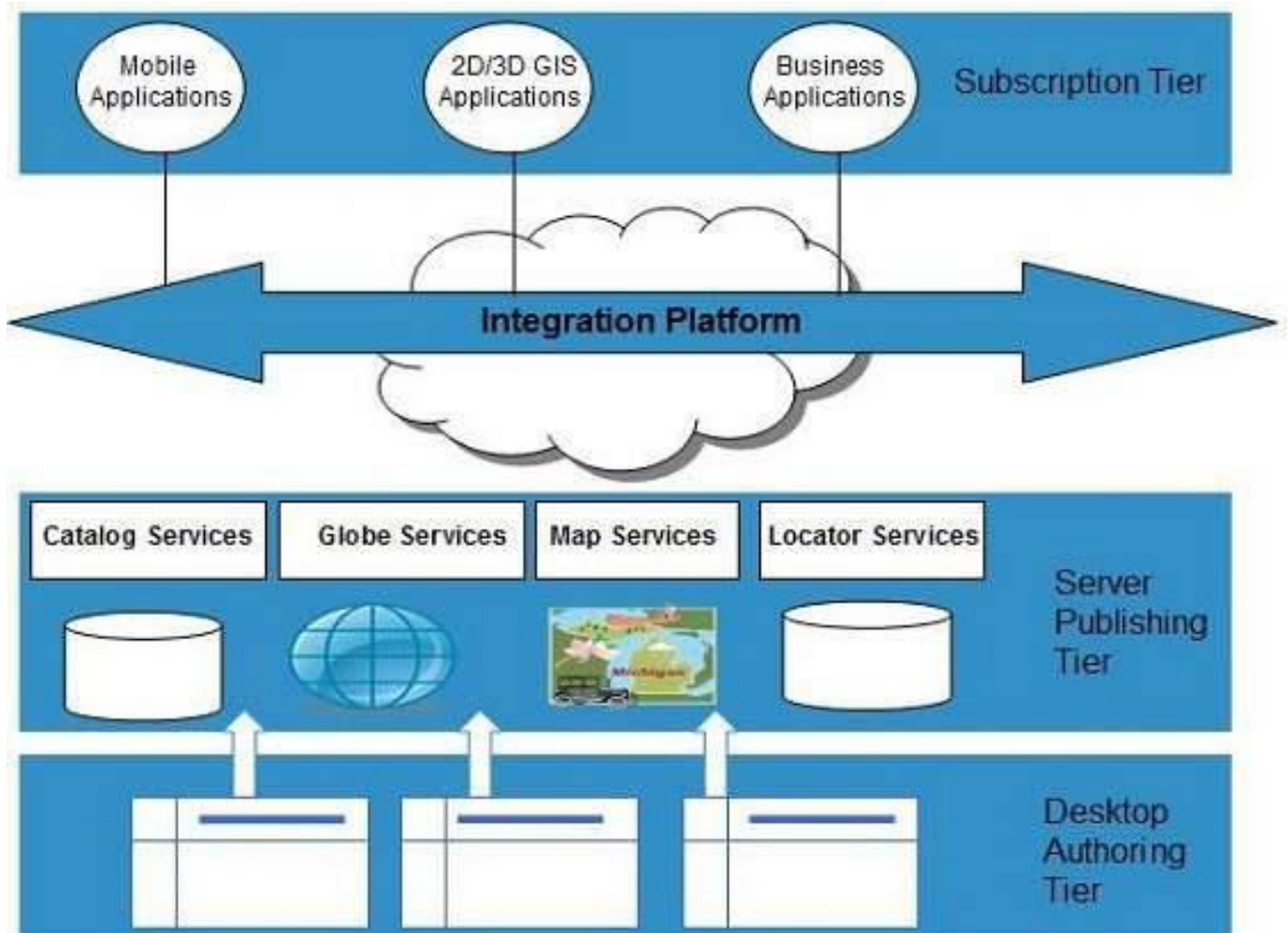


Fig.6

Grid Computing

Grid Computing refers to distributed computing, in which a group of computers from multiple locations are connected with each other to achieve a common objective. These computer resources are heterogeneous and geographically dispersed.

Grid Computing breaks complex task into smaller pieces, which are distributed to CPUs that reside within the grid.

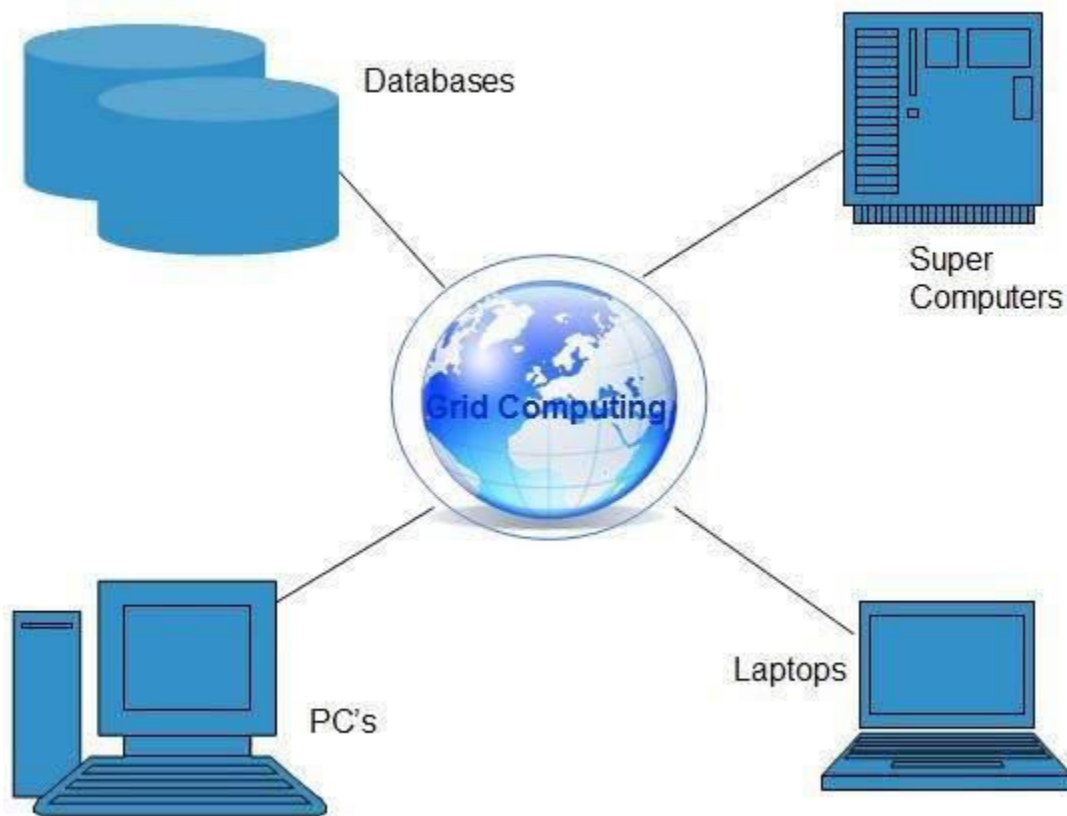


Fig.7

Utility Computing

Utility computing is based on **Pay-per-Use model**. It offers computational resources on demand as a metered service. Cloud computing, grid computing, and managed IT services are based on the concept of utility computing.

Advantages of Cloud Computing

There are various advantages of cloud computing technology. The important advantages of cloud computing are given below.

1) Lower cost computer for users

In cloud, you don't require a high-powered (and accordingly high-priced) computer to run cloud computing web based applications because applications run on cloud not on desktop PC or laptop.

2) Lower IT infrastructure cost

By using cloud computing, you need not to invest in larger numbers of more powerful servers, you also need not to require the IT staff for handling such powerful servers.

3) Fewer maintenance cost

The maintenance cost in cloud computing greatly reduces both hardware and software maintenance for organizations of all sizes.

4) Lower Software Cost

It reduces the software cost because you don't need to purchase separate software packages for each computer in the organization.

Disadvantages of Cloud Computing

There are various disadvantages of cloud computing technology. The important disadvantages of cloud computing are given below.

1) Require a constant Internet Connection

Cloud computing is impossible without Internet connection. To access any applications and documents you need a constant Internet connection.

2) Require High Speed Internet connection

Similarly, a low-speed Internet connection makes cloud computing painful at best and often impossible. Web based apps often require a lot of bandwidth to download, as need to download large documents.

3) Stored Data Might Not Be Secure

With cloud computing, all your data is stored in the cloud. That's all well and good, but how secure is the cloud? Can't unauthorized users gain access to your confidential data?

CHAPTER 4

4.1 Project Purpose and Scope

4.1 Purpose

The main aim of the project is used to derive a unique identity from the user's biometric data, which is further used to generate the user's private key.

4.2 Project Scope

In Proposed system, we introduced secure and efficient file storage and sharing cloud environment using Remote cloud. Mutual authentication is very important between two target device as the user and to the cloud server as mutual authentication makes the server to trust the cloud storage where cloud owner can store the data and from server end, server will verify the user credential to provide any service to that particular user.

Cloud contains the encrypted information along with the parameter related to that file. If any user request to the cloud server with some attribute. Based on the attribute cloud will redirect all the related file to the user in order to decrypt the file user need the key for that file for that user request to RSA and DSE(Key Generation Centre) with the file attribute and

4.3 Product Perspective

In the authentication phase, we capture a new biometric fingerprint image of the user, and subsequently generate the private key and encrypt the biometric data as a query. This

queried biometric data is then transmitted to the authentication server for matching with the stored data

4.4 System Features

From a captured user's fingerprint image, we extract all minutiae points. In order to increase the accuracy in feature extraction, we first align the fingerprint image. From this aligned fingerprint image, we select the consistent region. The consistent region can be defined as the fingerprint region, which has a high chance of appearance in any captured fingerprint image. We select this consistent region to extract the minutiae points. To select a set of minutiae points from the consistent region, we propose to use a horizontal segment. Horizontal segment is a small area of the consistent region, which has the highest number of minutiae points. We select these minutiae points to generate a Trellis diagram of the convolution coding [39] and finally, a codeword from it.

4.5 Design and Implementation Constraints

4.5.1 Constraints in Analysis

- ◆ Constraints as Informal Text
- ◆ Constraints as Operational Restrictions
- ◆ Constraints Integrated in Existing Model Concepts
- ◆ Constraints as a Separate Concept
- ◆ Constraints Implied by the Model Structure

4.5.2 Constraints in Design

- ◆ Determination of the Involved Classes
- ◆ Determination of the Involved Objects
- ◆ Determination of the Involved Actions
- ◆ Determination of the Require Clauses
- ◆ Global actions and Constraint Realization

4.5.3 Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object oriented methods.

4.6 Other Nonfunctional Requirements

4.6.1 Performance Requirements

The application at this side controls and communicates with the following three main general components.

- embedded browser in charge of the navigation and accessing to the web service;
- Server Tier: The server side contains the main parts of the functionality of the proposed architecture. The components at this tier are the following.

Web Server, Security Module, Server-Side Capturing Engine, Preprocessing Engine, Database System, Verification Engine, Output Module.

4.6.2 Safety Requirements

1. The software may be safety-critical. If so, there are issues associated with its integrity level
2. The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions.

3. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.
4. There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.
5. If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.
6. Systems with different requirements for safety levels must be separated.
7. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

CHAPTER 5

5.1 Architecture Diagram:

System Design:

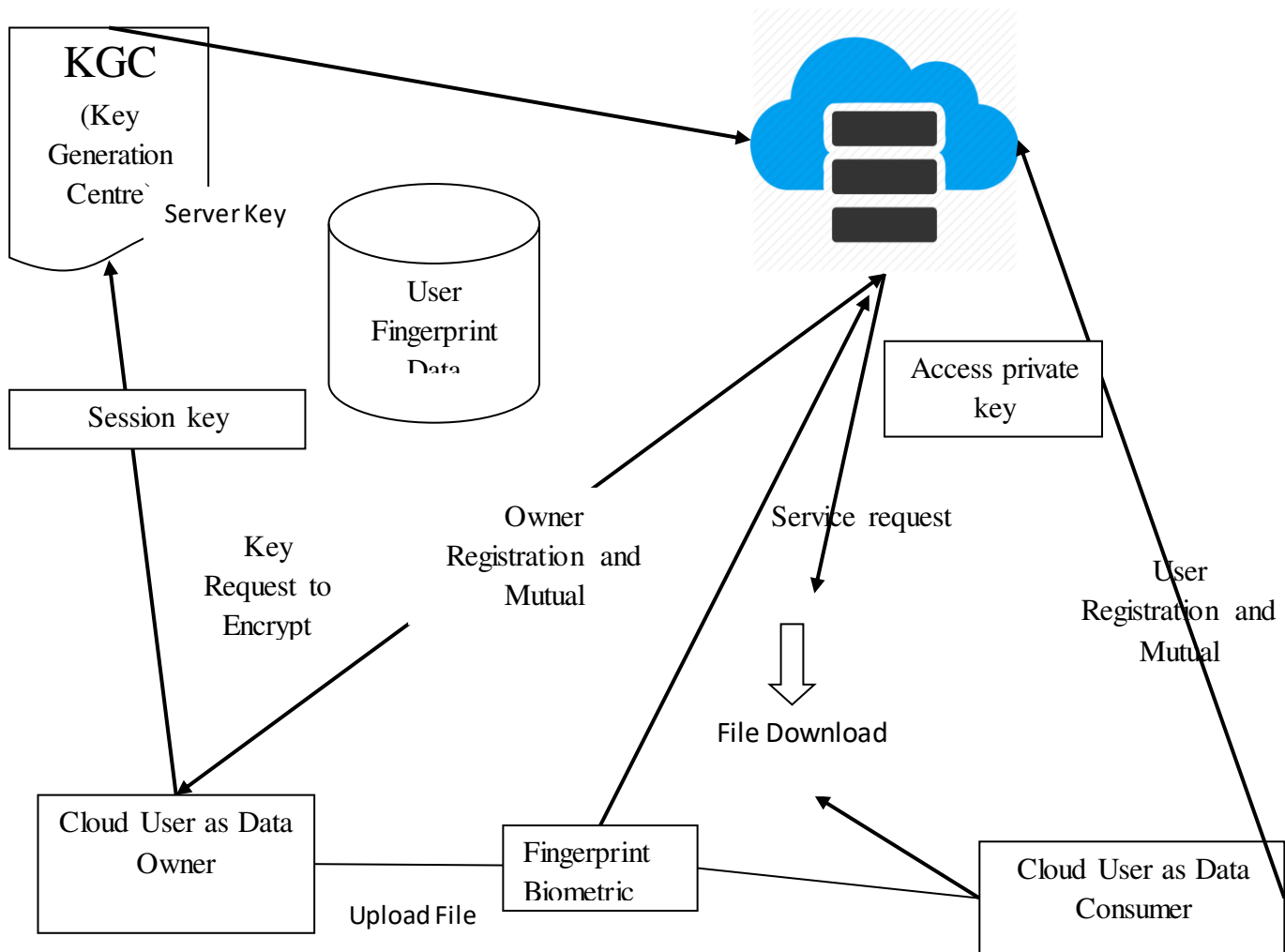


Fig: 8

5.2 Sequence Diagram:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

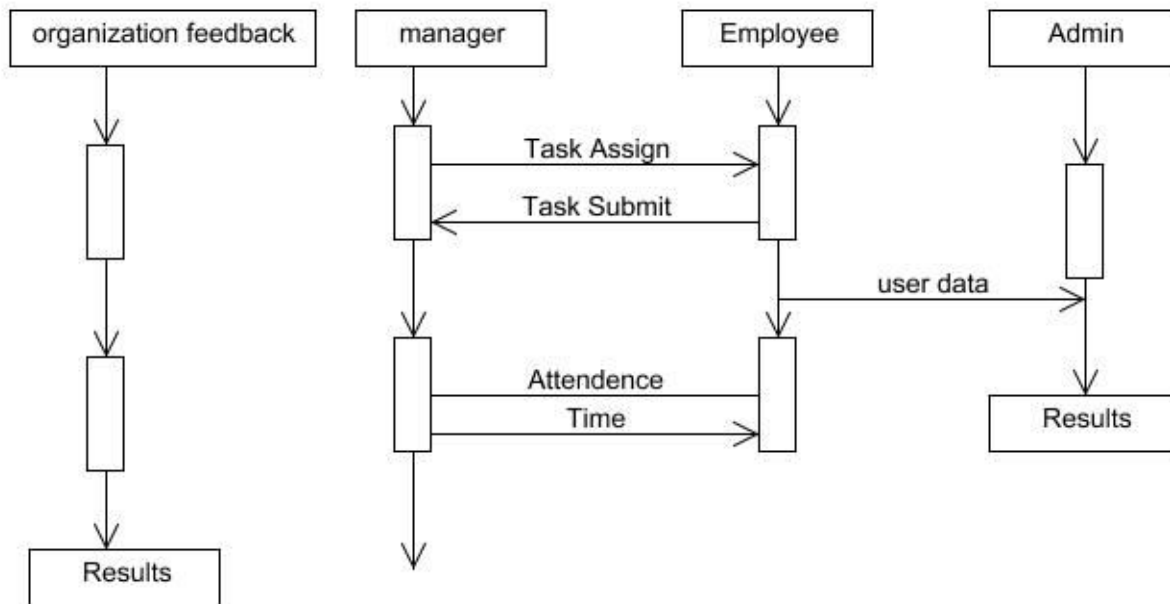


Fig: 9

5.3 Use Case Diagram:

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

5.3.1. USECASE DIAGRAM

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

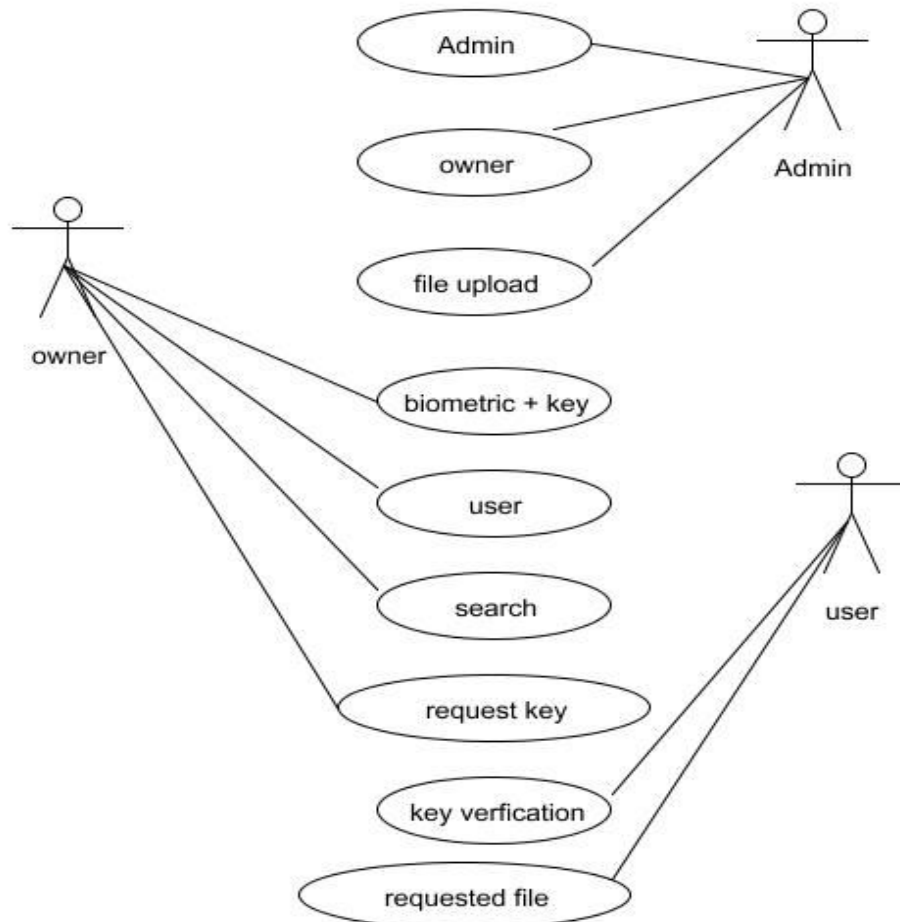


Fig: 10

5.4 Activity Diagram:

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.

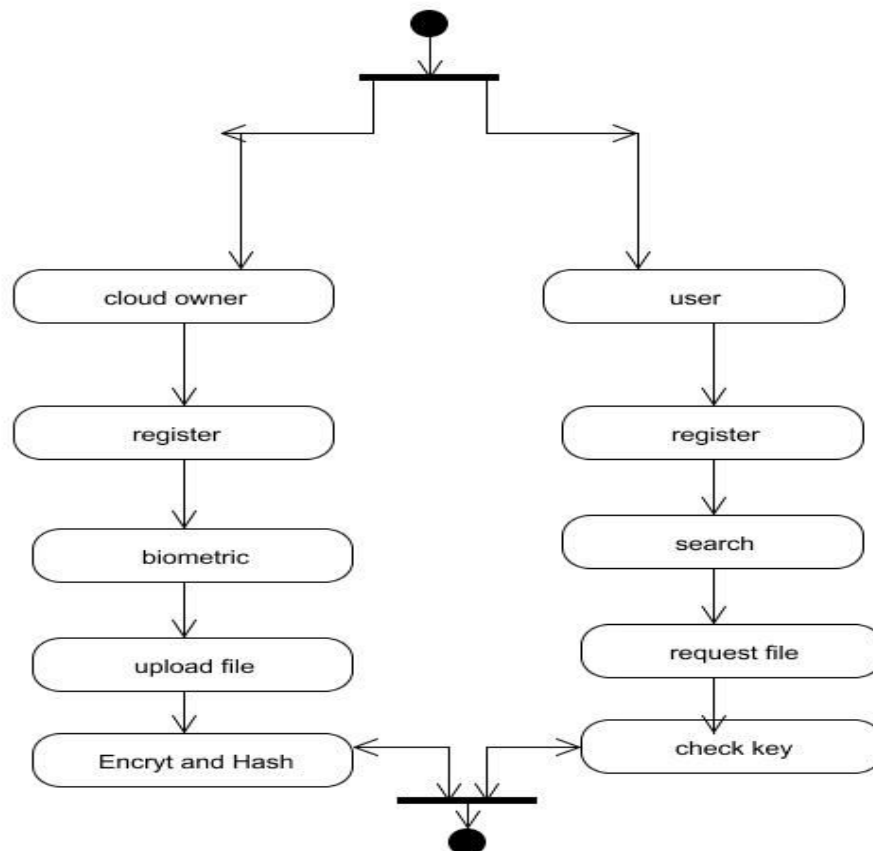


Fig: 11

5.5 Collaboration Diagram:

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and object

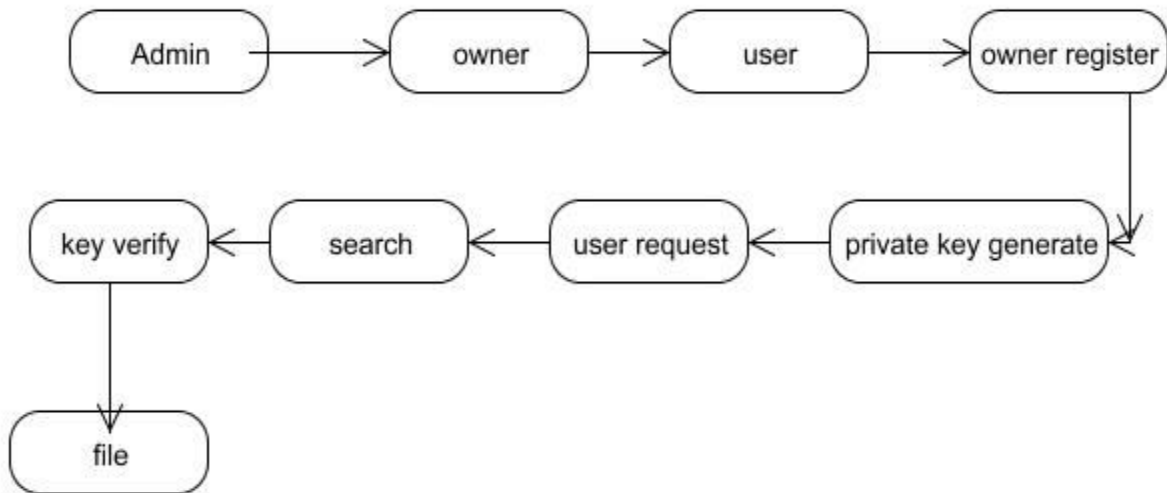
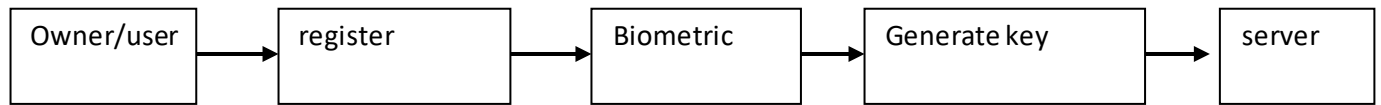


Fig: 12

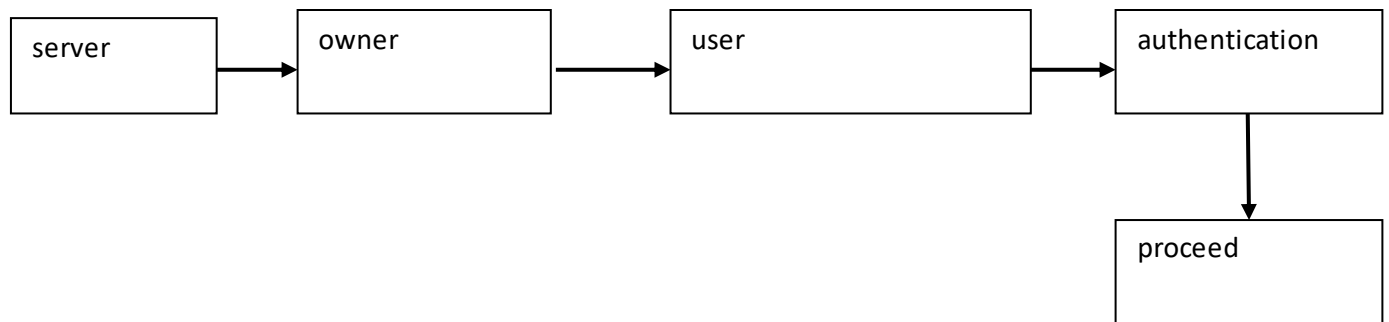
5.6 DATA FLOW DIAGRAM:

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview of the system which can later be elaborated DFDs can also be used for visualization of data processing.

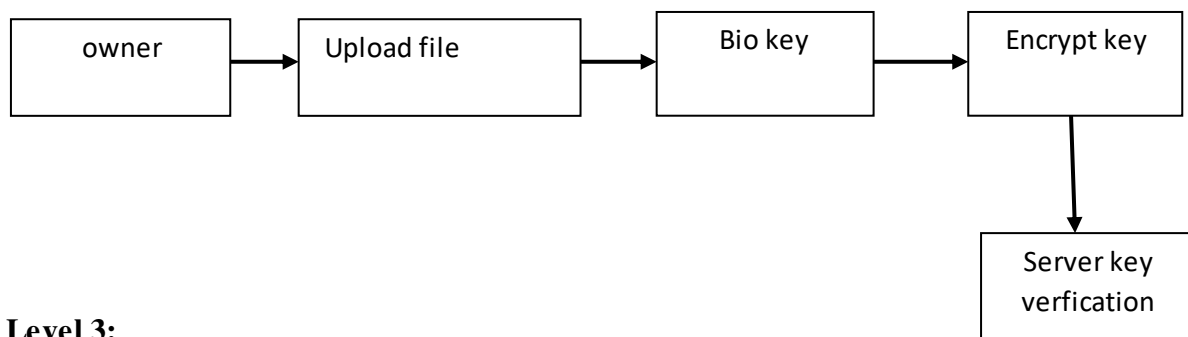
Level 0:



Level 1:



Level 2:



Level 3:

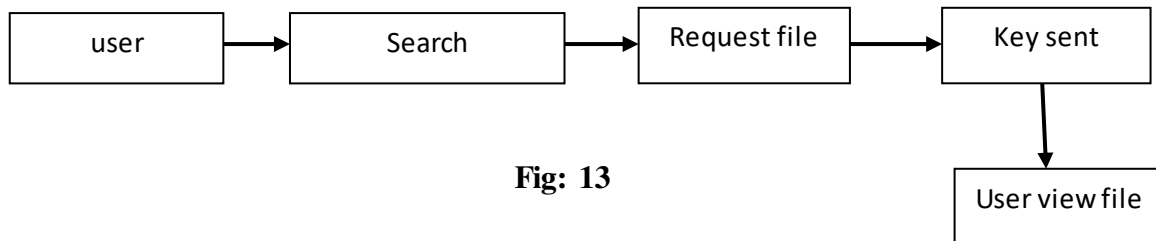


Fig: 13

5.7 Class Diagram

A Class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

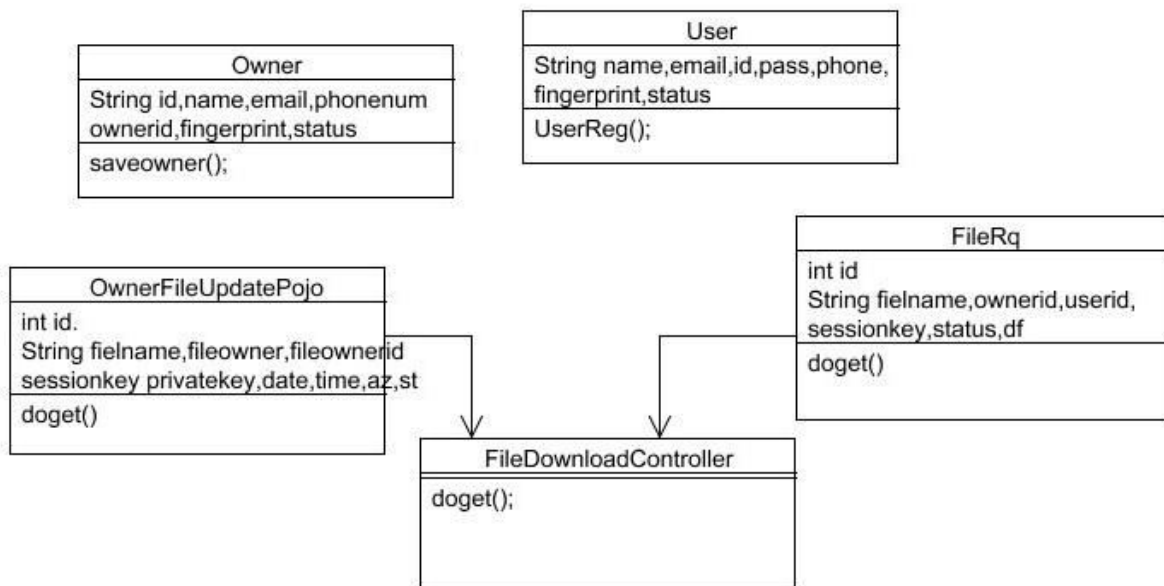


Fig: 14

CHAPTER 6

SYSTEM DESIGN

BioCAP has two key processes, namely: user registration and user authentication. The user registration requires private key generation, whereas user authentication requires generation of the session key and the message authenticator. BioCAP provides a provision to rollover the private key of a user. In addition, BioCAP is secure, computationally less expensive, and overcomes the inherent weaknesses of biometric verification. Moreover, BioCAP does not need pre-shared keys, and provides smooth mutual authentication mechanism and demands less number of keys to be managed from application and user point of view

6.1 MODULES

- Cloud Owner and User Registration & Key Generation
- Remote Server Authentication between Devices
- Cloud Owner Upload File to Remote Cloud Server
- Cloud User Decrypt and Download file

➤—

6.2 MODULE EXPLANATION:

6.2.1 Cloud Owner and User Registration & Key Generation

÷

Initially user need to create a account in the cloud sever by providing all the necessary information related to the user like email id, mobile number, age name, password and Cloud Owner a biometric fingerprint then server will generate a unique identity for that Owner and User. Once Owner And User successfully register then user request to Key Generation centre to provide the keys to user by providing the unique id that was generated by the server.

User can access the Cloud data only if he registers their name in key generation centre.

Cloud Owner Upload File to Cloud Server:

User can able to upload any file in the cloud in order to provide security to the file user need to encrypted the file in the cloud .Cloud contain the only the encrypted format information about the file and corresponding parameter related to the file. In order to maintain security user initially generated the parameter related to the file and along with the mode to upload the data mode as public or private if private then user need to specify the client or mention group of people who can access or download the file then based on these parameter RSA will generate a key and sent to the user based in these key user would encrypt the file and upload to server.

Decrypt and Download file:

———Then User would search the file based on the parameter if any parameter match in the server, then the server will load the entire related file to that parameter. Before download any file from the sever the user would also need to mutual authentication to the server, if user fingerprint authenticated to the server then he access the file as file is in encrypted format user need to decrypt the file to decrypt the file user request to KGC (Key Generation Centre) for key along with the file parameter and user attribute then user can able to download the file.

CHAPTER 7

CODING AND TESTING

7.1 CODING

Once the design aspect of the system is finalized the system enters into the coding and testing phase. The coding phase brings the actual system into action by converting the design of the system into the code in a given programming language. Therefore, a good coding style has to be taken whenever changes are required it easily screwed into the system.

7.2 CODING STANDARDS

Coding standards are guidelines to programming that focuses on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

Program should be simple, clear and easy to understand.

Naming conventions

Value conventions

Script and comment procedure

Message box format

Exception and error handling

7.2.1 NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The

conventions are adopted for **easy understanding** of the intended message by the user. So it is customary to follow the conventions. These conventions are as follows:

Class names

Class names are problem domain equivalence and begin with capital letter and have mixed cases.

Member Function and Data Member name

Member function and data member name begins with a lowercase letter with each subsequent letters of the new words in uppercase and the rest of letters in lowercase.

7.2.2 VALUE CONVENTIONS

Value conventions ensure values for variable at any point of time.

This involves the following:

- Proper default values for the variables.
- Proper validation of values in the field.
- Proper documentation of flag values.

7.2.3 SCRIPT WRITING AND COMMENTING STANDARD

Script writing is an art in which indentation is utmost important. Conditional and looping statements are to be properly aligned to facilitate easy understanding. Comments are included to minimize the number of surprises that could occur when going through the code.

7.2.4 MESSAGE BOX FORMAT

When something has to be prompted to the user, he must be able to understand it properly. To achieve this, a specific format has been adopted in displaying messages to the user. They are as follows:

- X – User has performed illegal operation.
- ! – Information to the user.

7.3 TEST PROCEDURE

SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

7.4 TEST DATA AND OUTPUT

7.4.1 UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

7.4.2 FUNCTIONAL TESTS

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

7.4.3 PERFORMANCE TEST

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

7.4.4 STRESS TEST

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

7.4.5 STRUCTURED TEST

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.
- Checking attributes for their correctness.

- Handling end of file condition, I/O errors, buffer problems and textual errors in output information

7.4.6 INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

7.5 TESTING TECHNIQUES / TESTING STRATERGIES

7.5.1 TESTING

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet –undiscovered error. A successful test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding an yet undiscovered error. A successful test is one that uncovers an yet undiscovered error. Any engineering product can be tested in one of the two ways:

7.5.1.1 WHITE BOX TESTING

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been design to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

- Flow graph notation
- Cyclometric complexity
- Deriving test cases
- Graph matrices Control

7.5.1.2 BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

- Graph based testing methods

- Equivalence partitioning
- Boundary value analysis
- Comparison testing

7.5.2 SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

- Testing begins at the module level and works “outward” toward the integration of the entire computer based system.
- Different testing techniques are appropriate at different points in time.
- The developer of the software and an independent test group conducts testing.
- Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

7.5.2.1 INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when we put them together. The problem of course, is “putting them together”- interfacing. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

7.5.2.2 PROGRAM TESTING:

The logical and syntax errors have been pointed out by program testing. A syntax error is an error in a program statement that in violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax error. These errors are shown through error messages generated by the computer. A logic error on the other hand deals with the incorrect data fields, out-off-range items and invalid

combinations. Since the compiler s will not deduct logical error, the programmer must examine the output. Condition testing exercises the logical conditions contained in a module. The possible types of elements in a condition include a Boolean operator, Boolean variable, a pair of Boolean parentheses A relational operator or on arithmetic expression. Condition testing method focuses on testing each condition in the program the purpose of condition test is to deduct not only errors in the condition of a program but also other a errors in the program.

7.5.2.3 SECURITY TESTING:

Security testing attempts to verify the protection mechanisms built in to a system well, in fact, protect it from improper penetration. The system security must be tested for invulnerability from frontal attack must also be tested for invulnerability from rear attack. During security, the tester places the role of individual who desires to penetrate system.

7.5.2.4 VALIDATION TESTING

At the culmination of integration testing, software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test-validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in manner that is reasonably expected by the customer. Software validation is achieved through a series of black box tests that demonstrate conformity with requirement. After validation test has been conducted, one of two conditions exists.

- * The function or performance characteristics confirm to specifications and are accepted.
- * A validation from specification is uncovered and a deficiency created.

Deviation or errors discovered at this step in this project is corrected prior to completion of the project with the help of the user by negotiating to establish a method for resolving deficiencies. Thus the proposed system under consideration has been tested by using validation

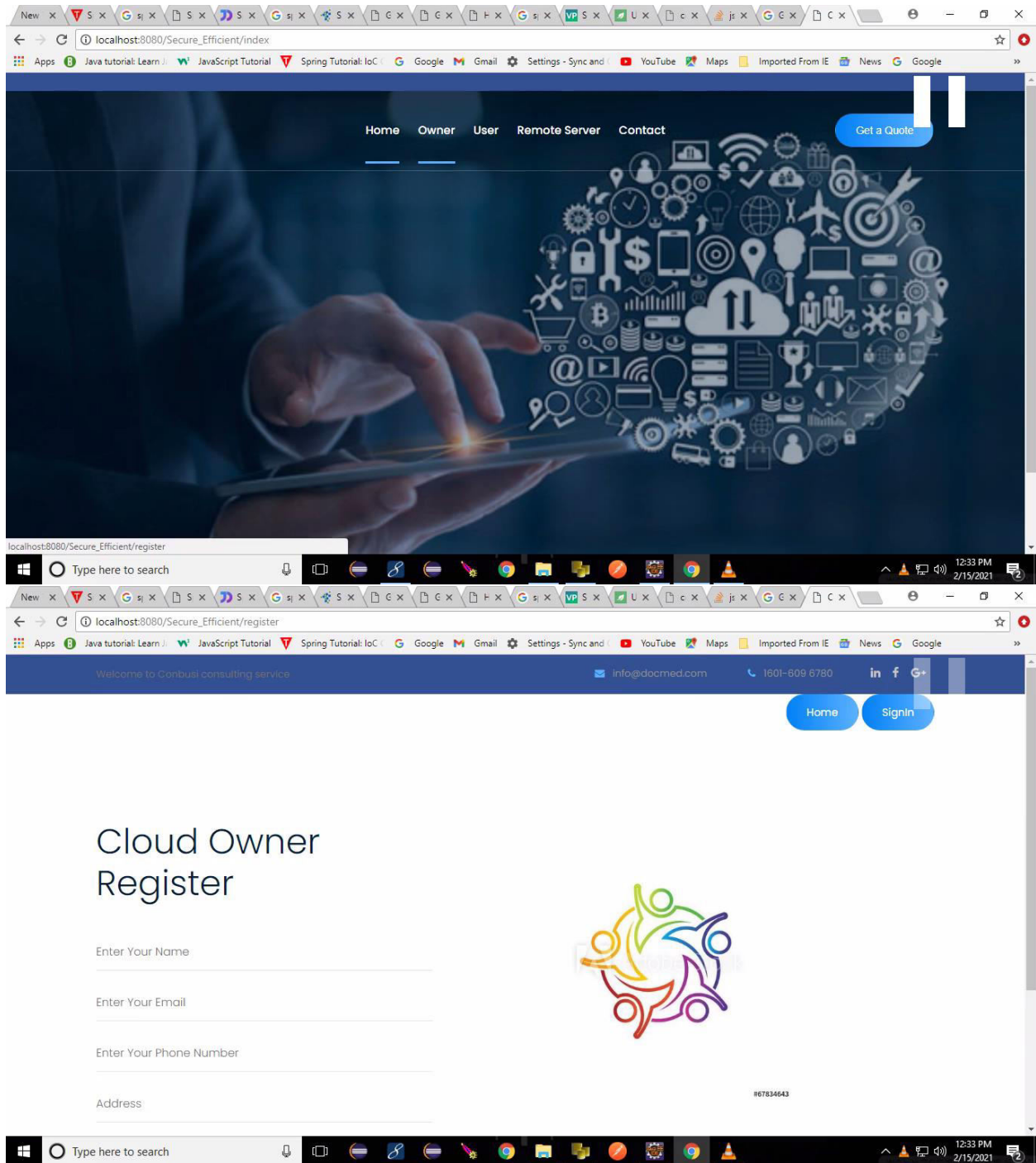
testing and found to be working satisfactorily. Though there were deficiencies in the system they were not catastrophic

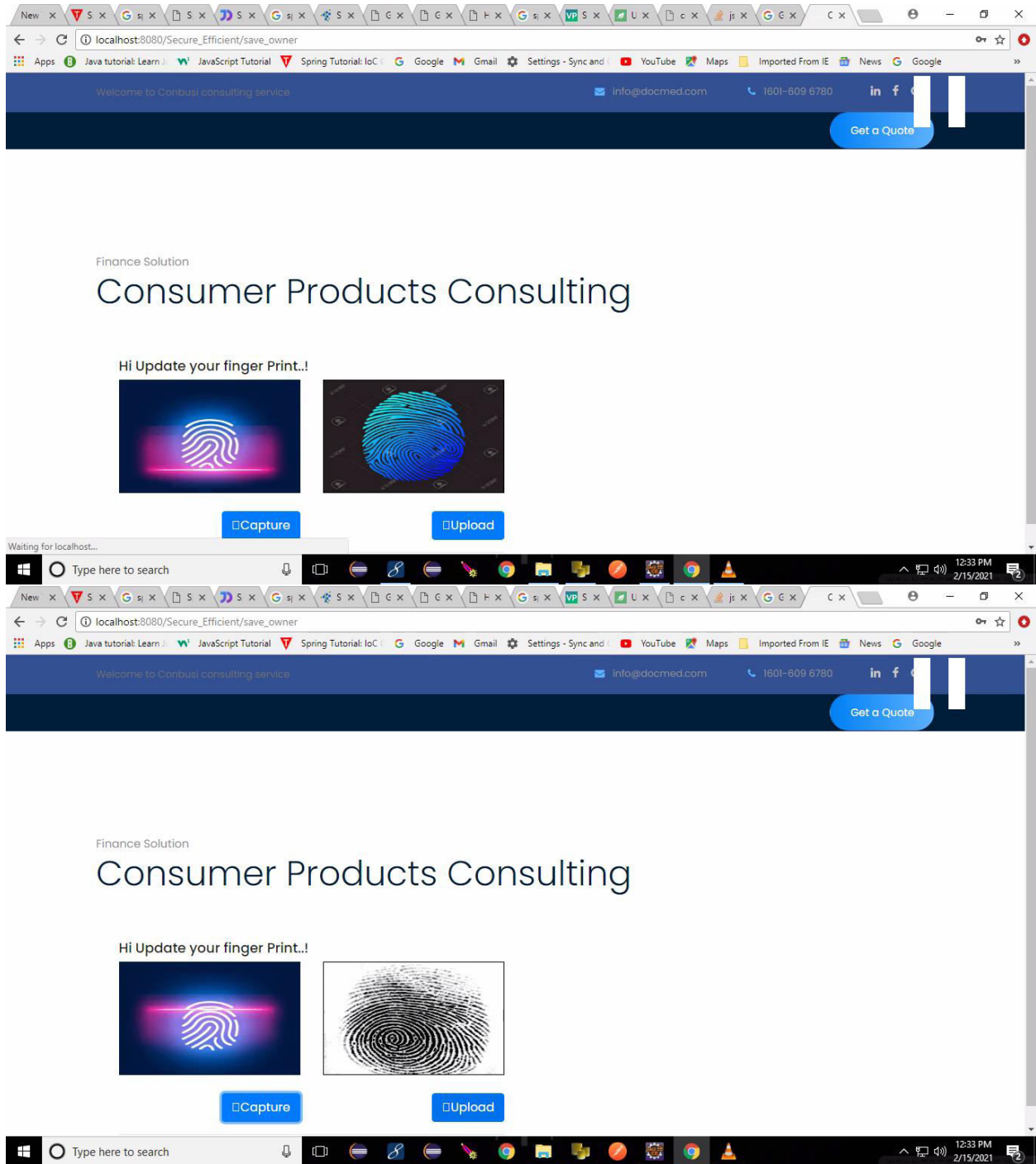
7.5.2.5 USER ACCEPTANCE TESTING

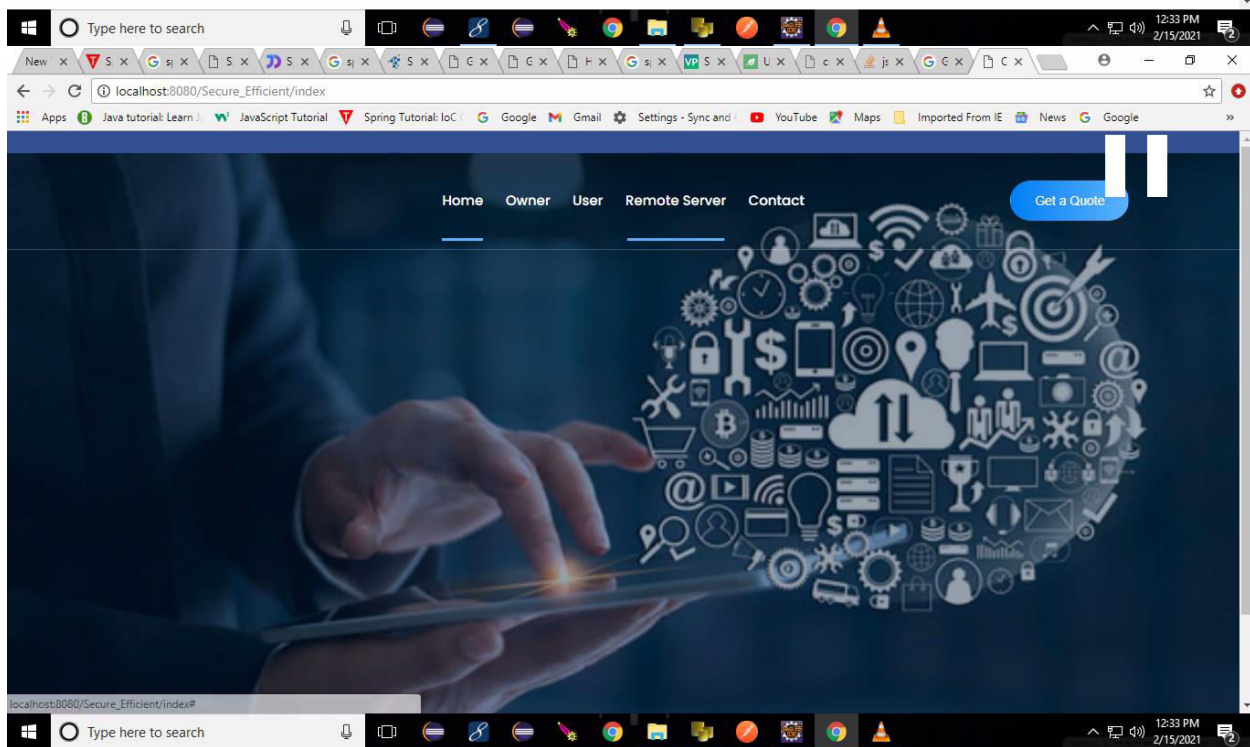
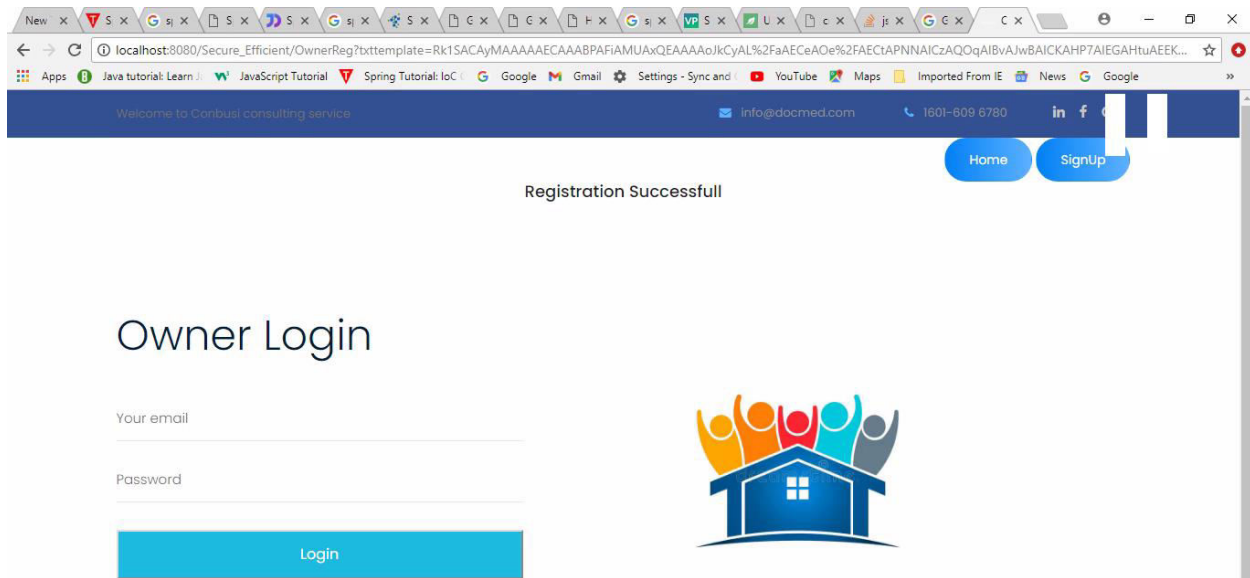
User acceptance of the system is key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system and user at the time of developing and making changes whenever required. This is done in regarding to the following points.

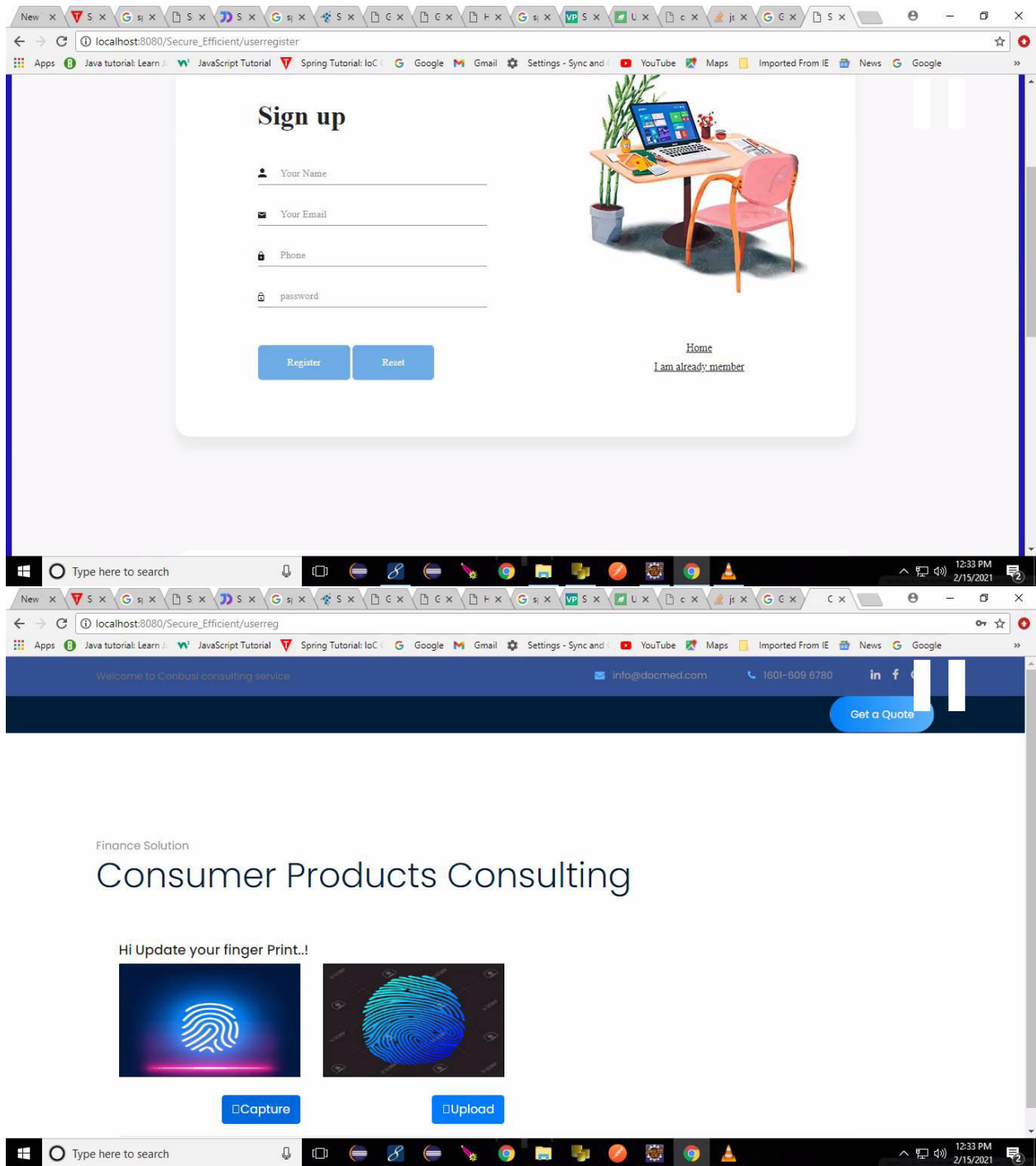
- Input screen design.
- Output screen design.

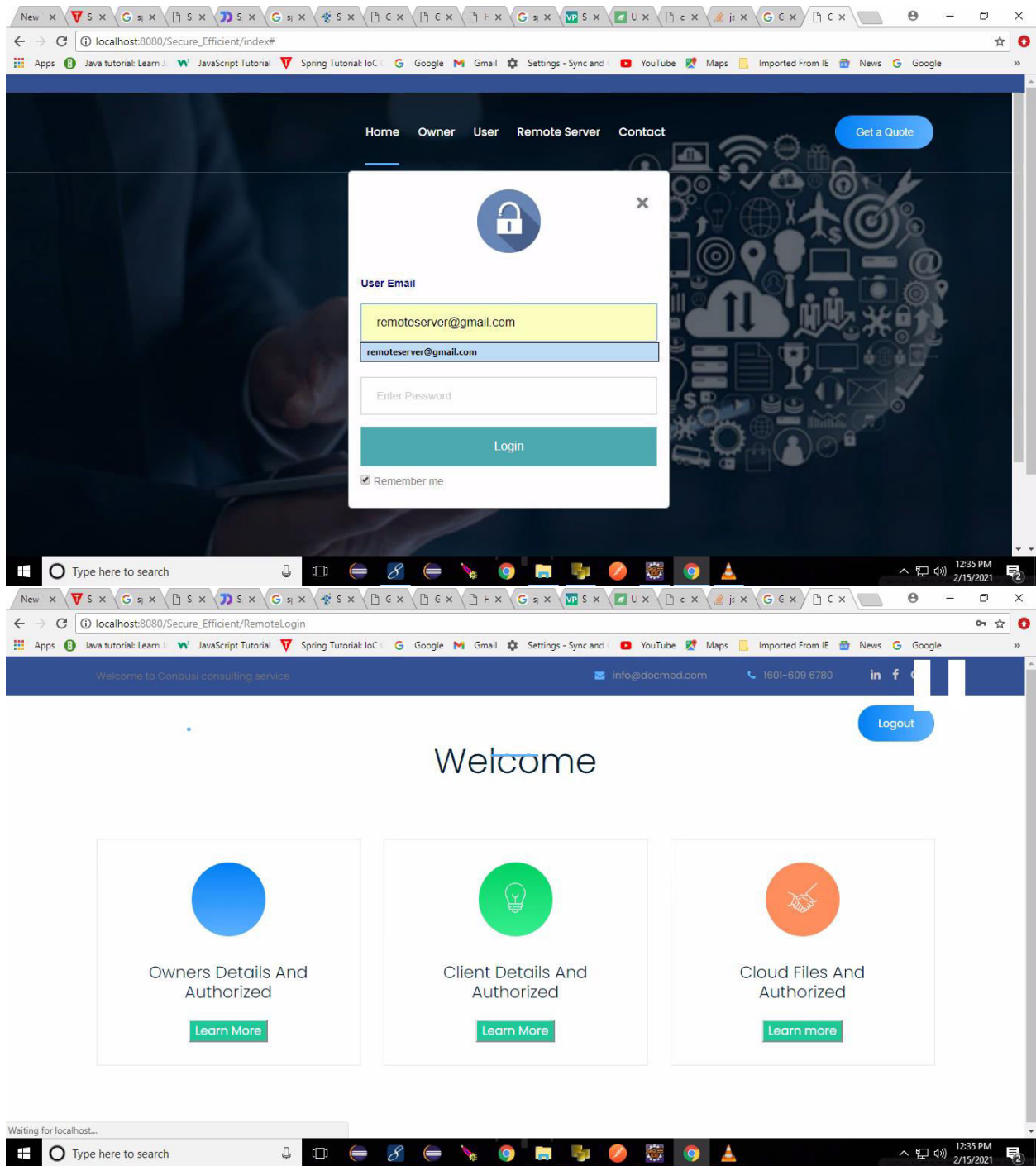
Screenshots:



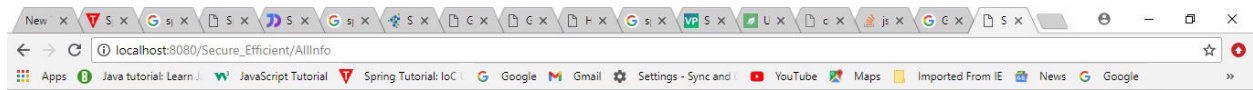










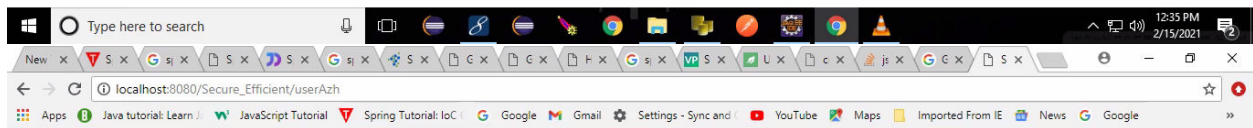


Cloud- User Information

Menu Logout

User List

Userid	Name	Email	phone	Authentication	Status
user207	kumar	kumar@gmail.com	9632540178	pending	



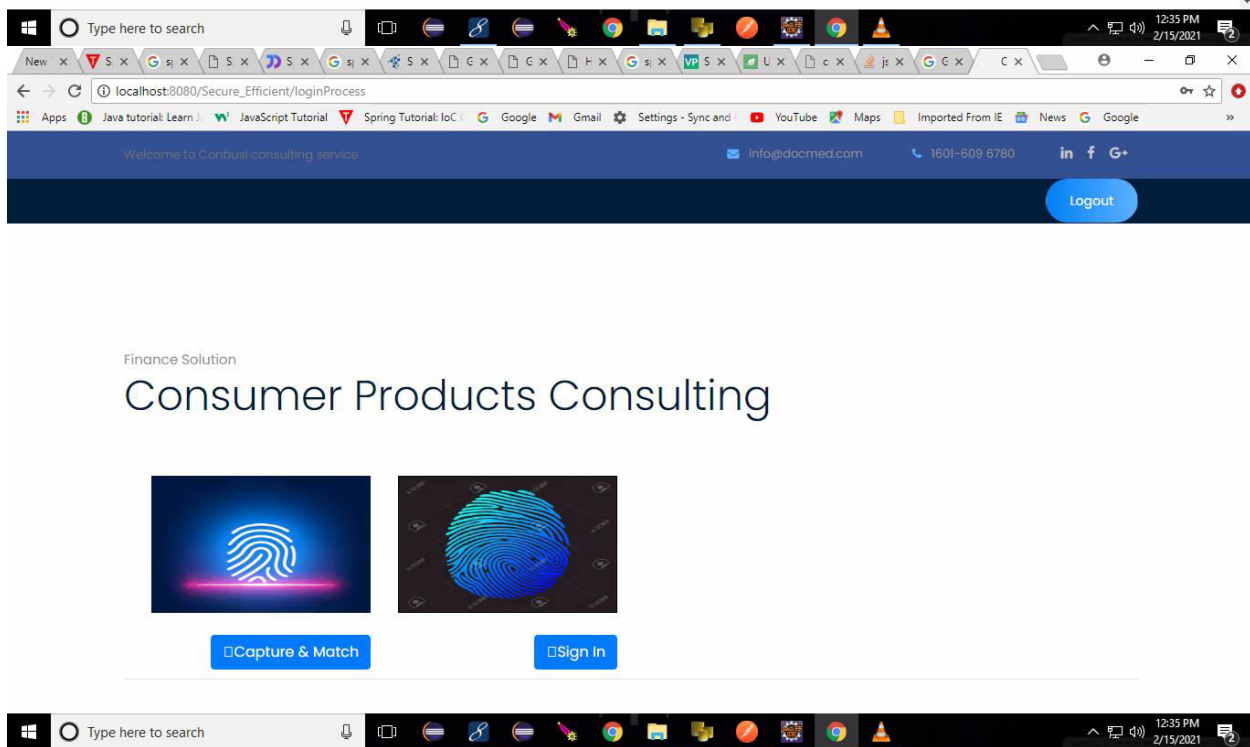
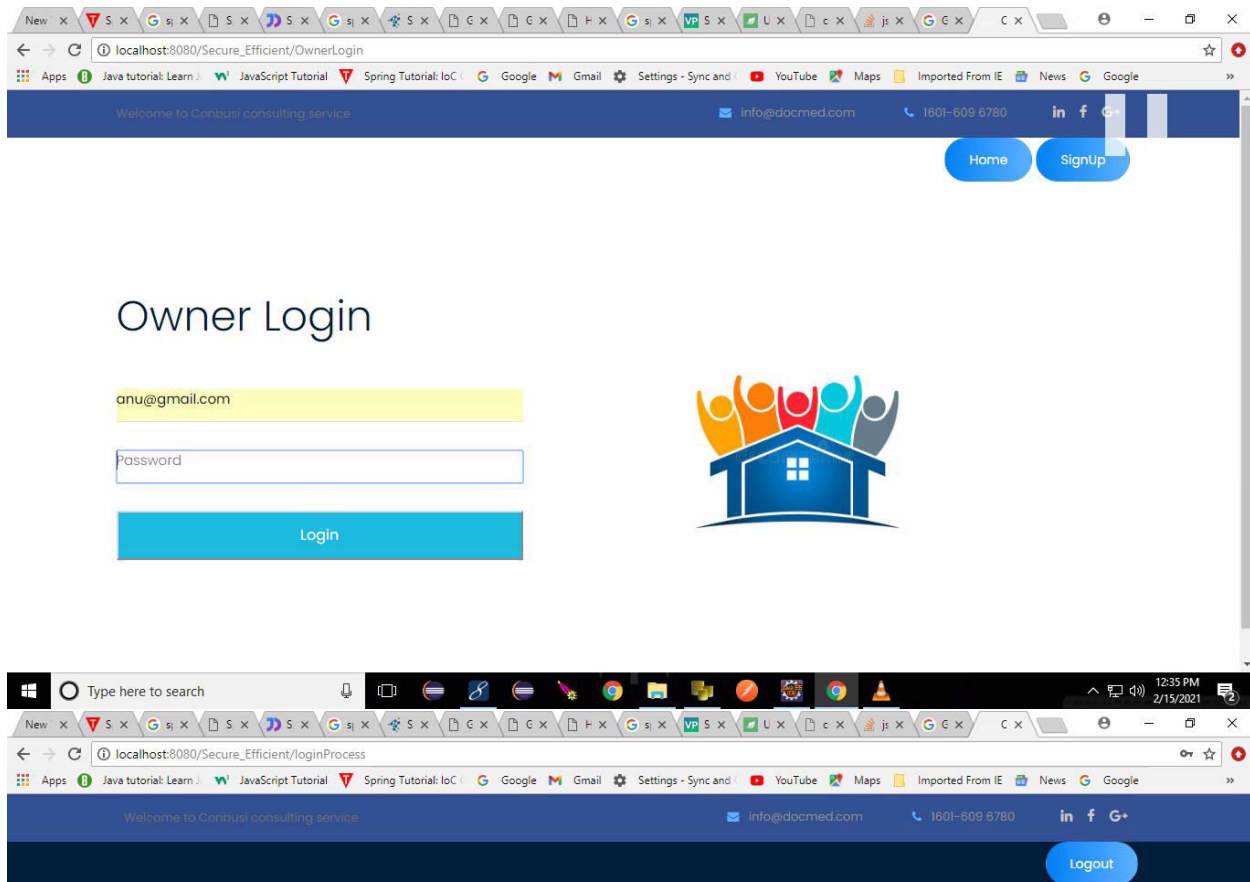
Cloud- User Information

Menu Logout

User List

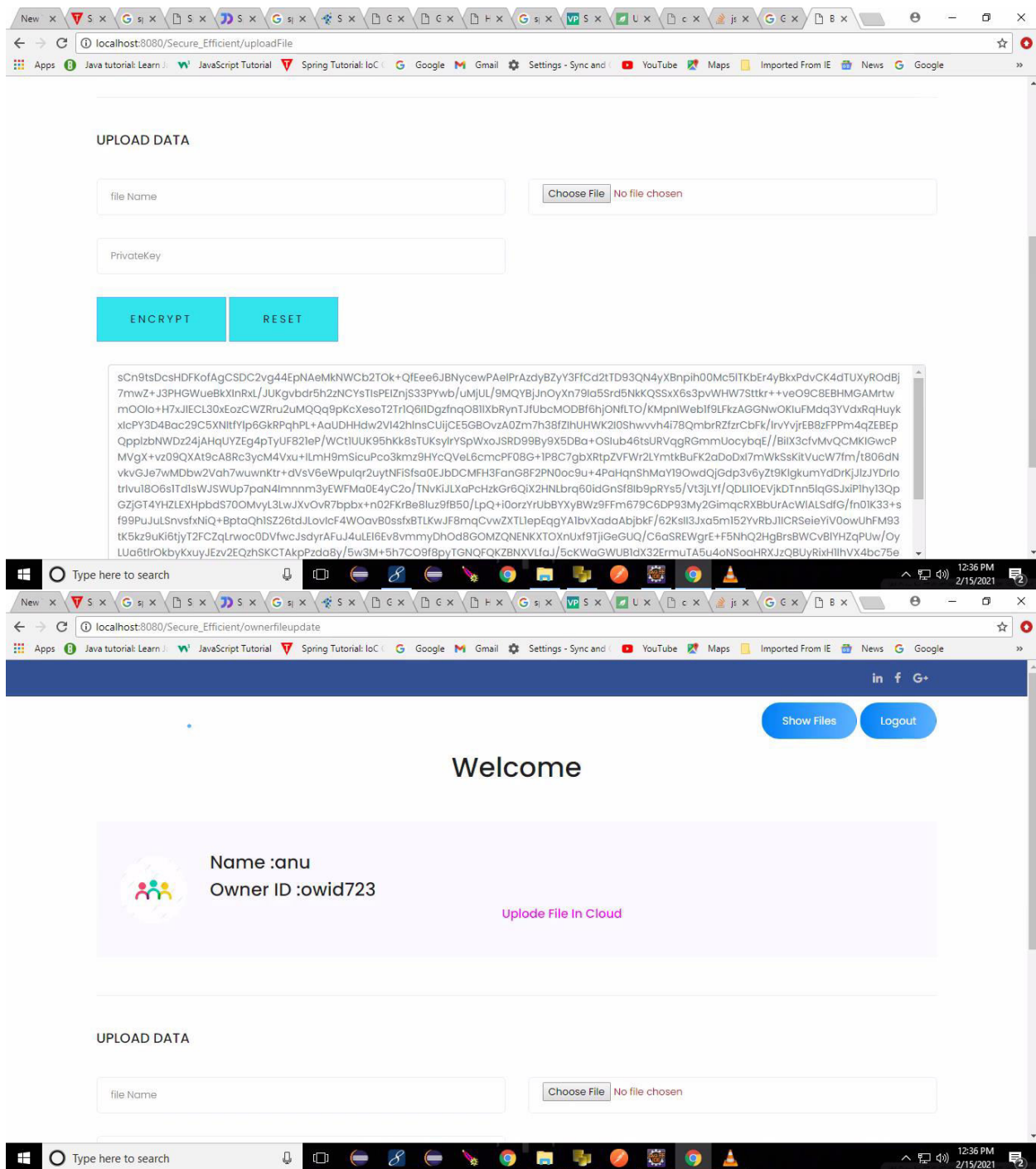
Userid	Name	Email	phone	Authentication	Status
user207	kumar	kumar@gmail.com	9632540178	Your Are Accepted	





Browser window showing a web application running on localhost:8080/Secure_Efficient/LoginSucess. The page displays user information: Name :anu, Owner ID :owid723. Below this is an "UPLOAD DATA" section with input fields for file name and private key, and buttons for ENCRYPT, RESET, and UPLOAD. The file name field contains "javadoc" and the private key field contains "7263053". The file upload button shows "Choose File" and "CreateGroupPojo.java".

Waiting for localhost...



localhost:8080/Secure_Efficient/OwnerShowUpdateFile

Welcome to Conbusi consulting service

info@docmed.com 1601-609 6780

[R_Q_Files](#) [Logout](#)

View All Cloud Files

Menu

S_No	File_Name	File_Owner_Name	File_OwnerID	Date_Time	PrivateKey	Access_Authorization	Status	Category	Permission
1	javadoc	anu	owid723	2021/02/15 12:36:12	7263053	no	Pending	upload	Processing

Waiting for www.w3schools.com...

Type here to search

localhost:8080/Secure_Efficient/userregister



Sign up

[Log in](#)

Home Or login with [f](#) [t](#) [G](#)

[Home](#)

[Create an account](#)

Welcome to Conbusi consulting service

info@docmed.com 1601-609 6780

Cloud Files Logout

Request Pass File Owner

File_Name

FileOwnerID

Send

Cloud Update Files

S_No	File_Name	File Owner_Name	File OwnerID	Date_Time
------	-----------	-----------------	--------------	-----------

Welcome to Conbusi consulting service

info@docmed.com 1601-609 6780

Enter PIN Logout

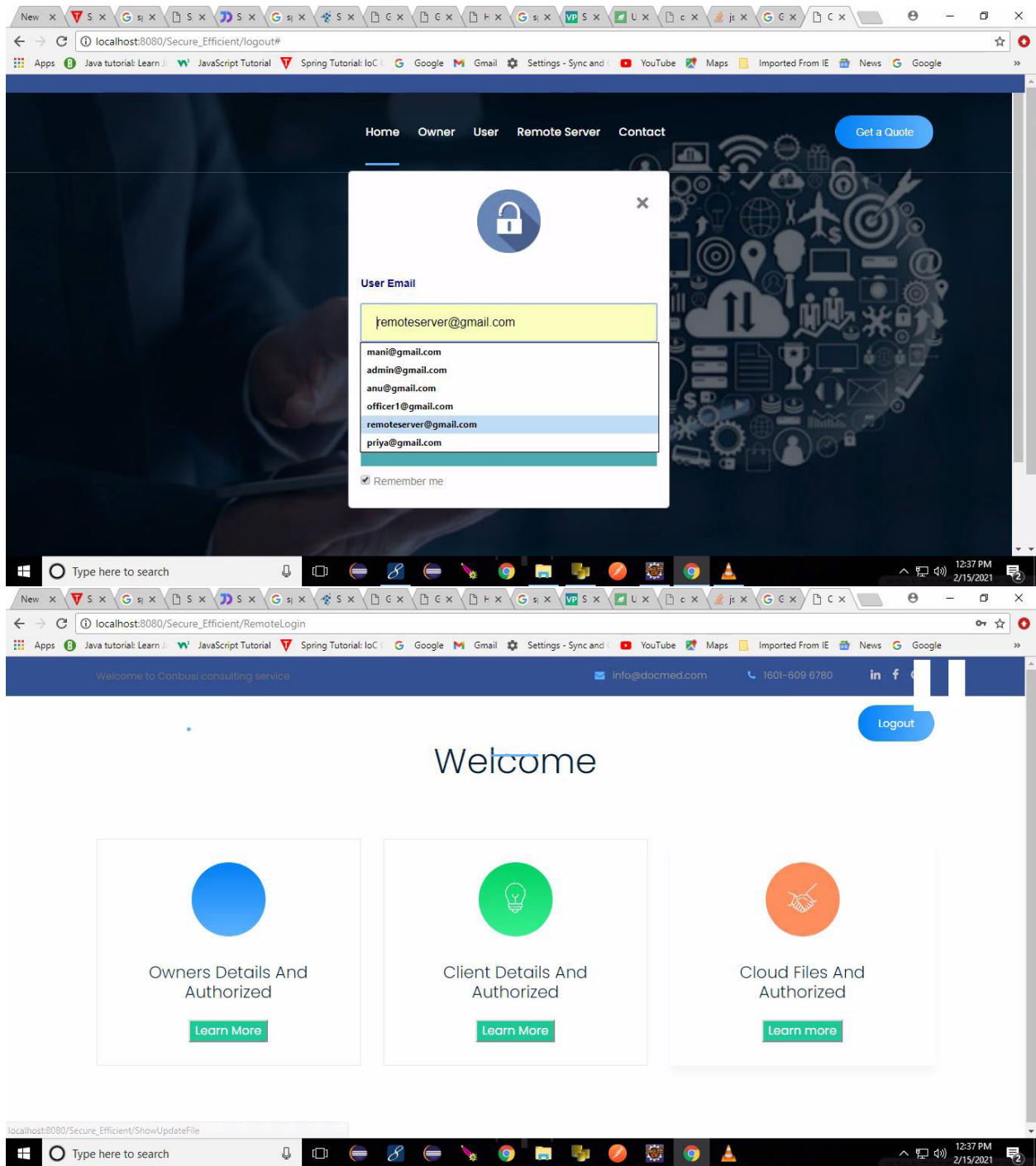
View All Cloud Files

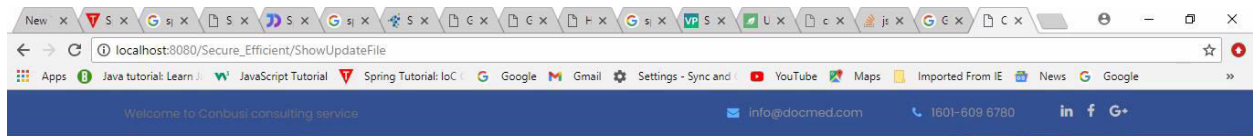
Menu

S_No	File_Name	File OwnerID	User_ID	Date_Time	Private Key	Status
No Records Present.						

Type here to search

12:37 PM 2/15/2021

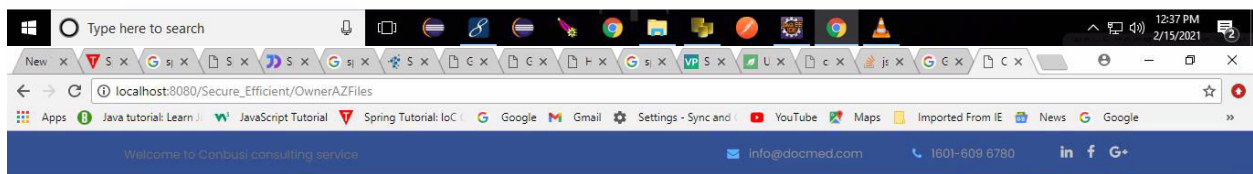




View All Cloud Files

Menu

S_No	File_Name	File_Owner_Name	File_OwnerID	Date_Time	Access_Authorization	Status	Category	Permission
1	javadoc	anu	owid723	2021/02/15 12:36:12	no	Pending	upload	click

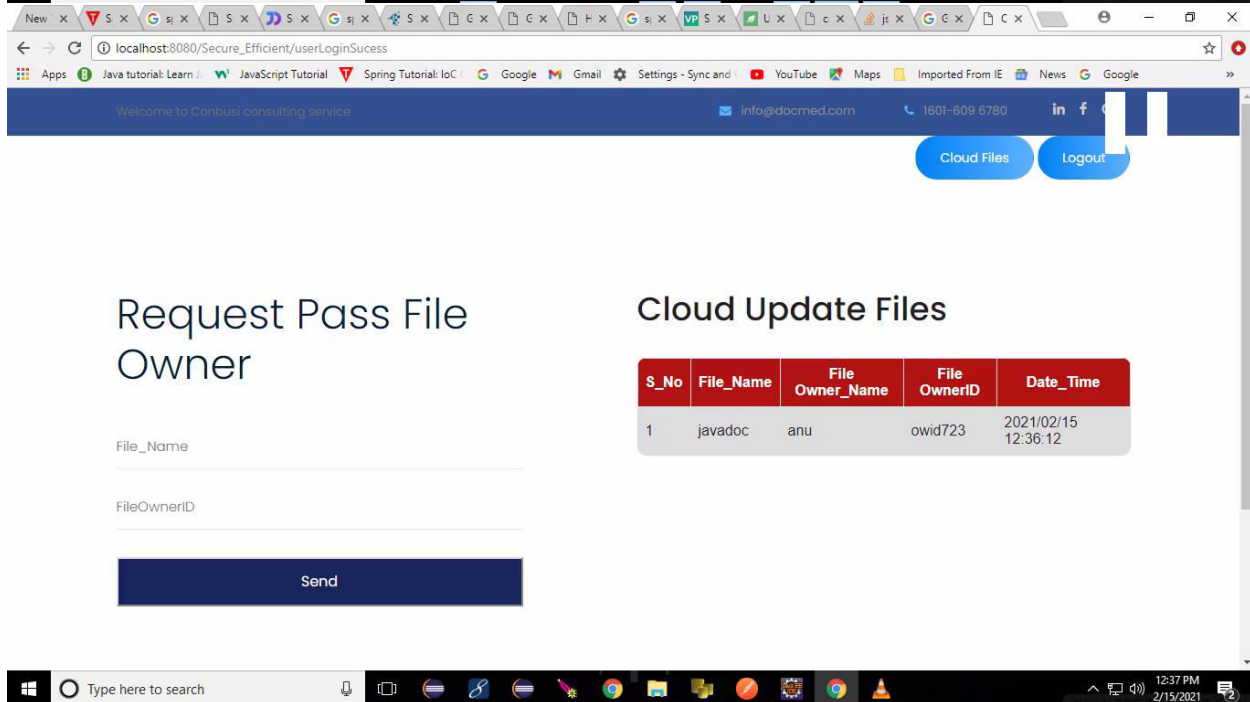
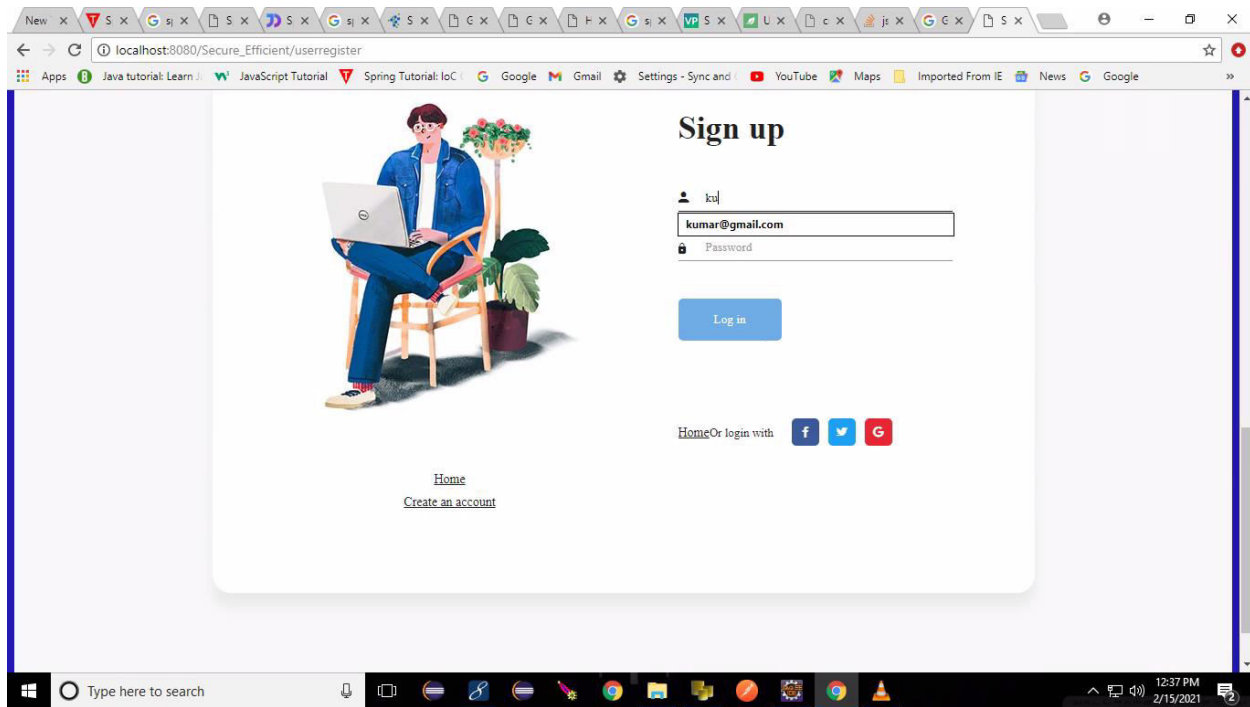


View All Cloud Files

Menu

S_No	File_Name	File_Owner_Name	File_OwnerID	Date_Time	Access_Authorization	Status	Category	Permission
1	javadoc	anu	owid723	2021/02/15 12:36:12	no	accepted	upload	Your Are Accepted!





Welcome to Corbus consulting service

info@docmed.com 1601-609 6780

Cloud Files Logout

Request Pass File Owner

javadoc

owid723

Send

Cloud Update Files

S_No	File_Name	File Owner_Name	File OwnerID	Date_Time
1	javadoc	anu	owid723	2021/02/15 12:36:12

Welcome to Corbus consulting service

info@docmed.com 1601-609 6780

Cloud Files Logout

File Request The Cloud Owner

Request Pass File Owner

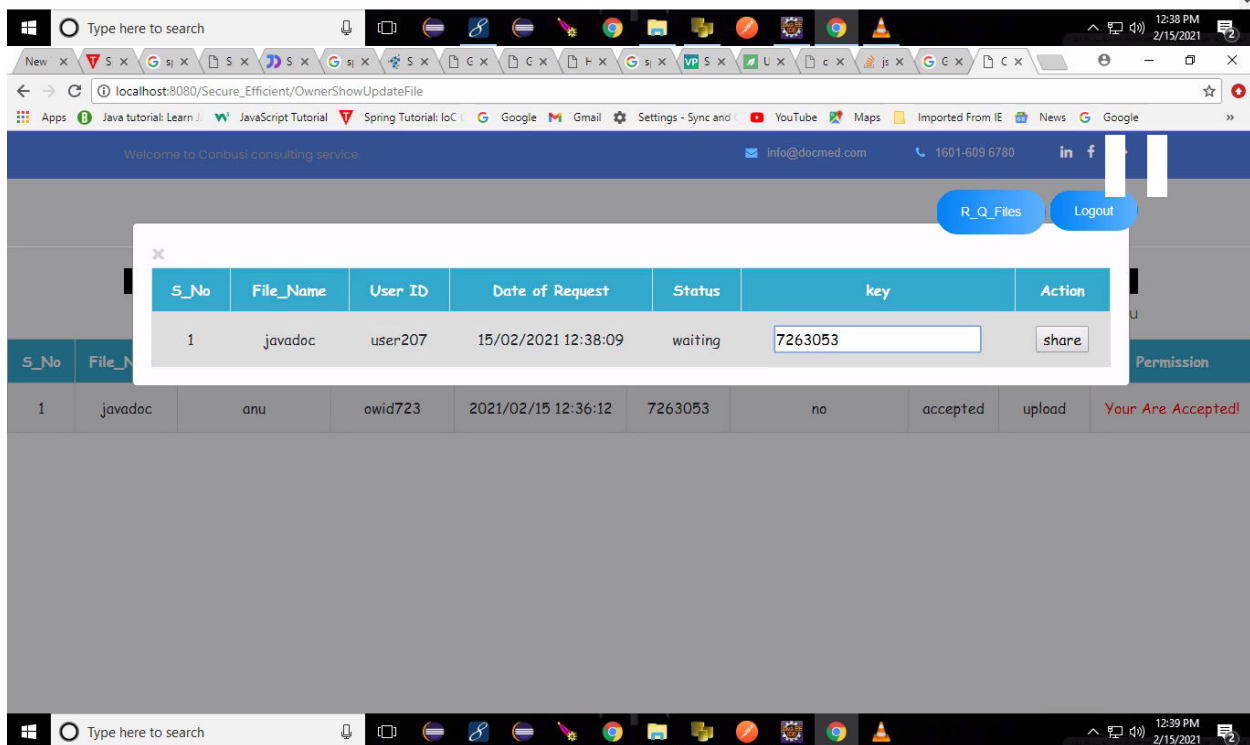
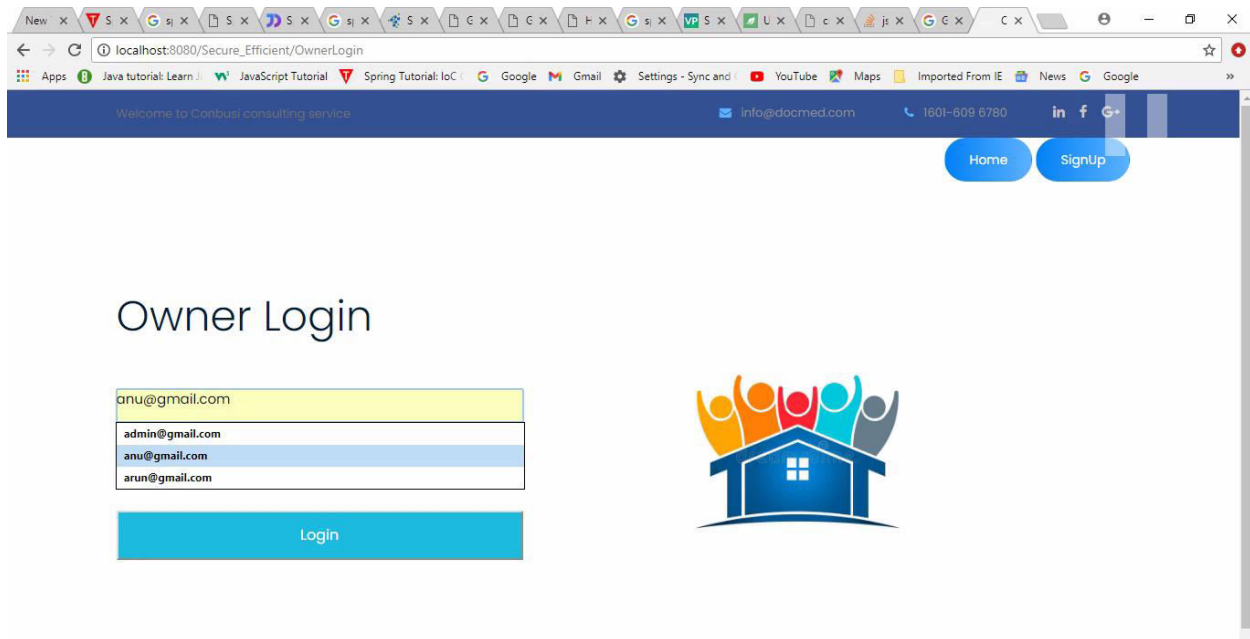
File_Name

FileOwnerID

Send

Cloud Update Files

S_No	File_Name	File Owner_Name	File OwnerID	Date_Time
------	-----------	-----------------	--------------	-----------





Welcome to Conbusi consulting service

info@docmed.com 1601-609 6780

Enter PIN Logout

Enter Your Key

7263053

Click

S_No	File_Name	File_OwnerID	_Time	Private Key	Status
1	javadoc	owid723	21 12:38:09	7263053	ok

localhost:8080/Secure_Efficient/rqFileDetails#

localhost:8080/Secure_Efficient/DownloadFile

encryption file

```
sCn9tsDcsHDFKofAgCSDC2vg44EpNAeMKNWCb2Tok
+QfEee6JBnycewPAelPrAzyBY3FfCd2tTD93QN4yX
Rozih0DMc5ITXbErdvBkvPdvCkAdtLlXvBndRI7mwwZ+13
```

File Decryption



File Decryption
Downloading File

encryption file

```
sCn9tsDcsHDFKofAgCSDC2vg44EpNAeMKNWCb2TOK  
+QfEee6JBnycewPAelPrAzdyBY3FfCd2tTD93QN4yX  
Rneih0M6c5ITXbFrAvBkvPdvCKAdTLiYvR0dRI7mowZ+13
```

File Decryption

```
groupmembers;  
  
public CreateGroupPojo() {  
  
}  
  
public int getId() {  
    return id;  
}
```

File Decryption
Downloading File

javadoc.txt
1,650 / 1,650 B

12:39 PM
2/15/2021

12:40 PM
2/15/2021

REFERENCES

- C. Neuman, S. Hartman, K. Raeburn, “The kerberos network authentication service (v5),” RFC 4120, 2005.
- [2] “OAuth Protocol.” [Online]. Available: <http://www.oauth.net/>
- [3] “OpenID Protocol.” [Online]. Available: <http://openid.net/>
- [4] G. Wettstein, J. Grosen, and E. Rodriguez, “IDFusion: An open architecture for Kerberos based authorization,” Proc. AFS and Kerberos Best Practices Workshop, June 2006.
- [5] A. Kehne, J. Schonwalder, and H. Langendorfer, “A nonce-based protocol for multiple authentications,” ACM SIGOPS Operating System Review, vol. 26, no. 4, pp. 84–89, 1992.
- [6] B. Neuman and S. Stubblebine, “A note on the use of timestamps as nonces,” Oper. Syst. Rev., vol. 27, no. 2, pp. 10–14, 1993.
- [7] J. Astorga, E. Jacob, M. Huarte, and M. Higuero, “Ladon : endto- end authorisation support for resource-deprived environments,” IET Infomration Security, vol. 6, no. 2, pp. 93–101, 2012.
- [8] S. Zhu, S. Setia, and S. Jajodia, “LEAP: efficient security mechanisms for large-scale distributed sensor networks,” Washington D.C., USA, October 2003, pp. 62–72.
- [9] A. Perrig, R. Szewczyk, D. Tygar, V. Wen, and D. Culler, “SPINS: security protocols for sensor networks,” ACM Wireless Networking, vol. 8, no. 5, pp. 521–534, 2002.
- [10] P. Kaijser, T. Parker, and D. Pinkas, “SESAME: The solution to security for open distributed systems,” Computer Communications, vol. 17, no. 7, pp. 501–518, 1994. 11] G. Wettstein, J. Grosen, and E. Rodriguez, “IDFusion: An open

architecture for Kerberos based authorization,” Proc. AFS and Kerberos Best Practices Workshop, June 2006.

[12] M. Walla, “Kerberos explained,” Windows 2000 Advantage Magazine, 2000.

[13] Q. Jiang, J. Ma, X. Lu, and Y. Tian, “An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks,” *Peer-to-Peer Networking and Applications*, vol. 8, no. 6, pp. 1070–1081, 2015.

[14] O. Althobaiti, M. Al-Rodhaan, and A. Al-Dhelaan, “An efficient biometric authentication protocol for wireless sensor networks,” *International Journal of Distributed Sensor Networks*, vol. 2013, pp. 1–13, 2013, Article ID 407971, <http://dx.doi.org/10.1155/2013/407971>.

[15] K. Xue, C. Ma, P. Hong, and R. Ding, “A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks,” *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 316 – 323, 2013.

[16] M. Turkanovic, B. Brumen, and M. Holbl, “A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the internet of things notion,” *Ad Hoc Networks*, vol. 20, pp. 96 – 112, 2014.

[17] M. Park, H. Kim, and S. Lee, “Privacy Preserving Biometric-Based User Authentication Protocol Using Smart Cards,” in *17th International Conference on Computational Science and Engineering*, Chengdu, China, 2014, pp. 1541–1544.

[18] P. K. Dhillon and S. Kalra, “A lightweight biometrics based remote user authentication scheme for IoT services,” *Journal of Information Security and Applications*, vol. 34, pp. 255 – 270, 2017.

[19] S. D. Kaul and A. K. Awasthi, “Security Enhancement of an Improved Remote User Authentication Scheme with Key Agreement,” *Wireless Personal Communications*, vol. 89, no. 2, pp. 621–637, 2016.

- [20] D. Kang, J. Jung, H. Kim, Y. Lee, and D. Won, “Efficient and Secure Biometric-Based User Authenticated Key Agreement Scheme with Anonymity,” *Security and Communication Networks*, vol. 2018, pp. 1–14, 2018, Article ID 9046064, <https://doi.org/10.1155/2018/9046064>.
- [21] D. Dolev and A. C. Yao, “On the security of public key protocols,” *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.
- [22] A. K. Das, “A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks,” *Peer-to-Peer Networking and Applications*, vol. 9, no. 1, pp. 223–244, 2016.
- [23] ———, “A secure and effective biometric-based user authentication scheme for wireless sensor networks using smart card and fuzzy extractor,” *International Journal of Communication Systems*, vol. 30, no. 1, pp. 1–25, 2017.
- [24] C. T. Li, C. Y. Weng, and C. C. Lee, “An advanced temporal credentialbased security scheme with mutual authentication and key agreement for wireless sensor networks,” *Sensors*, vol. 13, no. 8, pp. 9589–9603, 2013.
- [25] D. He, N. Kumar, and N. Chilamkurti, “A secure temporal-credentialbased mutual authentication and key agreement scheme for wireless sensor networks,” in *International Symposium on Wireless and pervasive Computing (ISWPC)*, Taipei, Taiwan, 2013, pp. 1–6.
- [26] M. Turkanovic and M. Holbl, “An improved dynamic password-based user authentication scheme for hierarchical wireless sensor networks,” *IR ELEKTROTECHNIKA*, vol. 19, no. 6, pp. 109 – 116, 2013.
- [27] R. Amin and G. P. Biswas, “A secure light weight scheme for user authentication and key agreement in multi-gateway based wireless sensor networks,” *Ad Hoc Networks*, vol. 36, pp. 58–80, 2016.

- [28] C.-C. Chang and N.-T. Nguyen, “An Untraceable Biometric-Based Multi server Authenticated Key Agreement Protocol with Revocation,” *Wireless Personal Communications*, vol. 90, no. 4, pp. 1695–1715, 2016.
- [29] Z. Xia, C. Yuan, R. Lv, X. Sun, N. N. Xiong, and Y. Shi, “A Novel Weber Local Binary Descriptor for Fingerprint Liveness Detection,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018, doi: 10.1109/TSMC.2018.2874281.
- [30] C. Yuan, X. Sun, and Q. M. J. Wu, “Difference co-occurrence matrix using BP neural network for fingerprint liveness detection,” *Soft Computing*, vol. 23, no. 13, pp. 5157–5169, 2019.
- [31] W. Yang, S. Wang, J. Hu, G. Zheng, and C. Valli, “Security and Accuracy of Fingerprint-Based Biometrics: A Review,” *Symmetry*, vol. 11, no. 2, 2019. [Online]. Available: <https://www.mdpi.com/2073-8994/11/2/141>
- [32] X. Huang, X. Chen, J. Li, Y. Xiang, and L. Xu, “Further Observations on Smart-Card-Based Password-Authenticated Key Agreement in Distributed Systems,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1767–1775, 2014.

